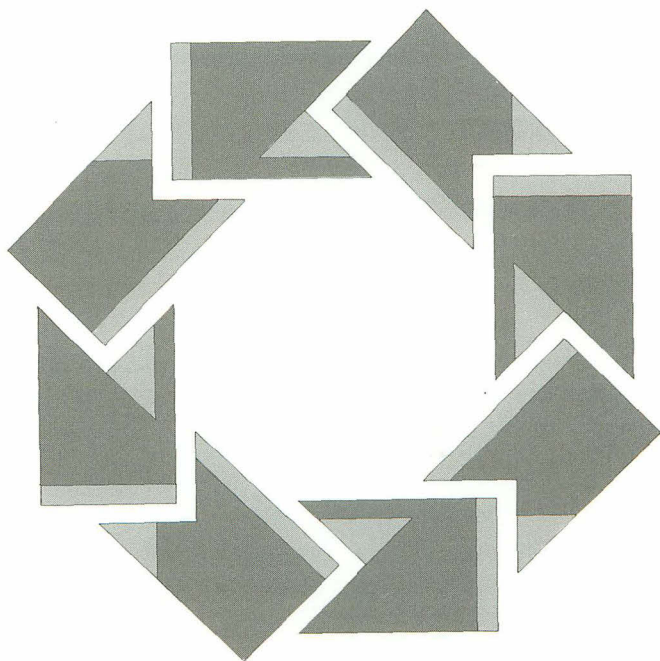
98ハードに強くなる本 II

ハードウェア・テクニカルガイド

東工大電算機愛好会&小高輝真 著

技術評論社

98ハードに強くなる本Ⅱ

ハードウェア・テクニカルガイド

東工大電算機愛好会&小高輝真 著

技術評論社

はじめに

ユーザがプログラムを開発するときは，ハードウェアの性能を生かした，使い勝手のよいプログラムを作りたいと思うものです．

PC-9801に付属しているN88 BASICは，ハードウェアの機能の大部分を比較的簡単に利用できるように設計されています．しかし，高速処理の必要などで機械語のプログラムを作る場合にはPC-9801の内部事情を知らなければプログラムを作ることができません．

また，近年MS-DOS上の言語でプログラムを開発する機会が増えていますが，MS-DOS上の言語ではOS上で他機との互換性を維持するためにPC-9801固有の機能（例えばグラフィックスなど）を扱うことが不得意なものがあります．また，文字の表示が遅かったりして，望みの性能を持ったプログラムを作ることが困難になることもあります．もちろん，OS上の互換性ということも時として大切ではありますが，普通はPC-9801のみで動けばよいプログラムを作ることの方が多いでしょう．このような時には，PC-9801内のBIOS，LIO等を利用すれば，グラフィックスなども比較的容易に活用できますし，ハードウェアを直接アクセスすることで，プログラムの高速化を図ることができます．

PC-9801は，数多くのハードウェアユニットから構成されているため，その構造を理解しようとして後込みしてしまった方もいるかもしれません．しかし，ユニット毎にばらして，それぞれを少しずつ理解してゆけば，PC-9801の全体像を摑むことはそう難しいことではありません．

なお，本書を理解するにあたっては機械語についての知識が必要となりますが，本書の目的はPC-9801のハードウェアを解説することですので機械語についての解説は割愛されています．PC-9801の機械語を習得されたい方は，「98マシン語」（藤木文彦著/技術評論社）という絶好の人気書がありますので，そちらを併読してください．

前書「PC98ハードに強くなる本」は1986年10月に出版されましたが，その後PC-9801シリーズはさらに性能を強化した新機種がいくつも登場しました．そこで，新機種の新たな機能をフォローすべく内容を改訂・強化し，「PC98ハードに強くなる本II」として出版することになりました．

本書でPC-9801のハードウェアを理解し，よりよいソフトウェアの開発に役立てていただきたいと思います．

1988年2月　　　　小高輝真

第1章　ハードウェアの知識

Contents

Contents

第6章　インターフェースと周辺機器

第7章　UX & VX

第1章

ハードウェアの知識

CONTENT

1 概要

第１章では，PC-98の基本的な内部構造について解説します．

PC-98*の持つ機能は多彩で，それだけ内部構造も複雑です．本体の外蓋をはずしてみるとわかるように，各種の機能を備えたたくさんのLSIが，複雑なネットワークの上に載って，１つのシステムを形成しています（人間の体で言えば，脳や心臓などの器官が無数の神経や血管で連絡しているのと同様です）．

これらの各LSIの器官について解説していくのが，本章の目的です．

以下にその概要を示します．

１．概要

２．システム構成とBIOS

PC-98のシステム構成について解説するとともに，本書で重点的に取り上げたBIOS**（基本入出力ルーチン）について，その使い方を中心に基本的な説明を加えています．BIOSを利用することにより，各種ハードウェアの制御が簡単な手続きで実現できるようになります．N_{88}-BASICでは未活用のままになっているハードウェアの能力を利用したりする際に，このBIOSの使い方を知っていると，とても便利です．BIOSはハードウェアを切り離して考えることができないので，各LSI解説の項で具体的な使用例を示しています．

３．CPU（中央処理装置）

CPUはPC-98のシステムを構成するLSI全体を制御する中心的LSIで，コンピュータの中枢です．このCPUがシステムの性能を大きく左右すると言ってよいでしょう．PC-98では16ビットCPUとしてμPD8086（E/F/M），V30***（U/UV/VF/VM/VX/UX），i80286（VX/UX）を搭載し，システムの優れた性能を引き出しています．

---

*　PC-98とはPC-9801の種々のバージョンを総称する意味で用いている表現である．本書では9801E/F/1, 2, 3/M2, 3/U2/UV2, 21/VF2/VM0, 2, 4, 21/VX0, 2, 4, 01, 21, 41/UX21, 41について取り扱っている．バージョンの違いを明確にする必要がある場合には，その都度明記する．

**　BIOS（Basic Input Output System）ハードウェアに密着している制御プログラムである．

***　i8086と上位コンパチブル．正式名称はμPD70116．

4．I／Oポート

PC-98には，各LSIごとにI／Oポートという入出力制御用のポートが設けられていて，各種インターフェースをはじめとするLSIを直接制御する際に重要な役割を果たします．

5．割り込みコントローラ（PIC）*

割り込みコントローラは，CPUの制御機能を補助するLSIです．各LSIは，CPUから送られる命令を受けて，それに対する応答をCPUに送り返しますが，CPUはそれらの全てを一度には処理できません．そこで，割り込みコントローラは，各々の応答に優先順位をつけて高位のものから順にCPUに送り出す働きをしています．

6．キーボード（KB）

キーボードから1本のカール・コードを介して本体に送られる信号がどのように変換されているのかを示すとともに，キーボードのI/O制御命令，キーボードKB BIOSの使い方を具体例を挙げて解説します．

7．タイマ

タイマはPC-98の脈拍数を決定するもの，つまりCPUを中心としていろいろな周辺装置へデータを送信する速さやタイミングを決定します．ここではタイマの動作について解説し，いくつかの応用例を示しています．

8．カレンダ時計

PC-98の内部には万年暦が入っています．ここでは，このカレンダ時計について説明し，内部データの表現形式や日付・時刻を設定したり読み出したりする方法について解説します．

---

* PIC(Programable Interrupt Controller)

2 システム構成とBIOS

PC-98は，CPUを中心として各種LSI*が集まってシステムを形成しています．この多彩な機能を持つシステムを使いこなすためには，システムがどのようなデバイスからなり，どのような機能を有するのかを知ることが重要です．そこでまず，システム構成の概要について示し，PC-98の全体像をつかむことにします．

また，実際にシステムからある機能を引き出すときには，各LSIが複雑に関連してきます．PC-98では，これらLSIを効率的に制御するために各LSIに固有のBIOS（基本入出力ルーチン）が用意されており，プログラム作成の上で非常に有用です．ここでは，BIOSについても簡単に触れておきます．

2.1 システム構成

図1-1にPC-98のシステム構成の概要を示します．

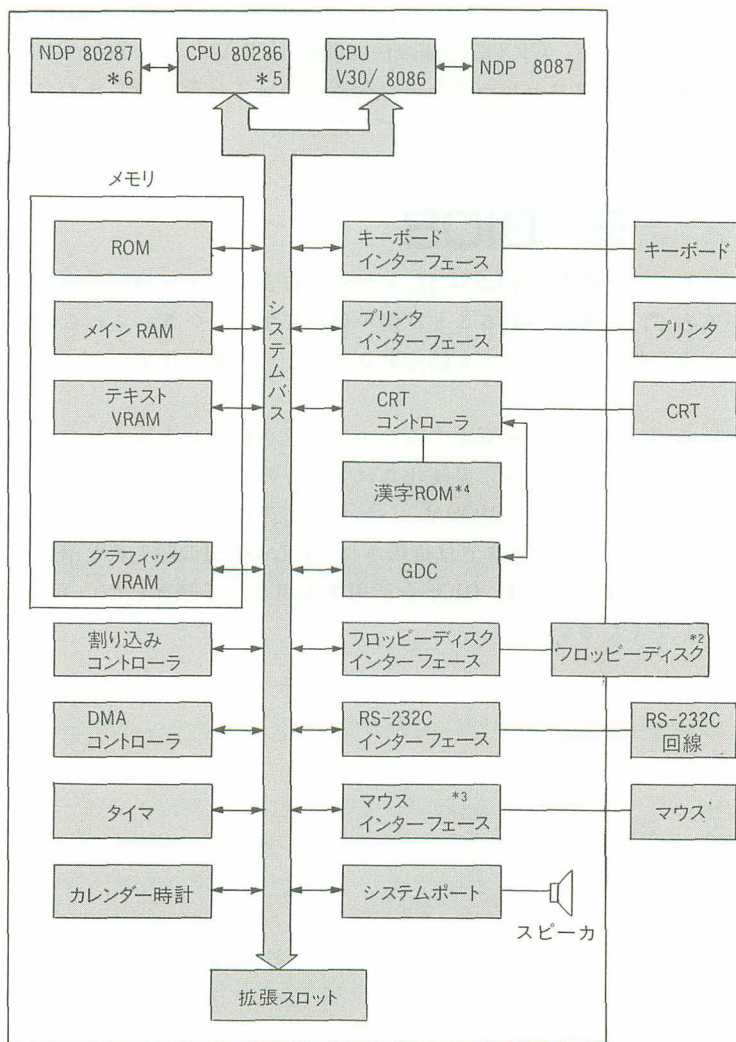
CPUは，演算処理とシステム全体の制御を行います．コ・プロセッサであるNDP（数値演算プロセッサ）を除く全てのLSIは，システムバスを介してCPUとつながっています．CPUは16ビットですが，周辺LSIの大部分は８ビットです．16ビットCPUでも従来製品を利用できるように設計されているので，このような構成が可能であるわけです．

CPUがある命令を出すと，それはシステムバスを通ってすべてのLSIに送られます．各LSIは，それが自分に対する命令であるかどうかを調べた後に実行します．逆に複数のLSIからCPUにデータが送られると，CPUの方では一度には処理できないので，割り込みコントローラ（PIC）というLSIがそれらの信号の優先順位を決定して，１つずつCPUに送ります．

こうした方式を取ることにより，CPUの割り込み制御に関する負荷が減り，相対的に処理速度が向上し，システムをコンパクトにまとめることが可能にな

---

* ＰＣ-98は，CPUを中心にして，多数の周辺装置が有機的に結合されたシステムである．CPUと周辺装置との間のインターフェース役を，これら各種ＬＳＩが果している．

図1-1　システム構成図

＊1　オプション
＊2　PC-9801E/VM0は内蔵フロッピーディスクなし
＊3　PC-9801E/F1/F2はオプション
＊4　PC-9801Eでは第１水準漢字ROMはオプション
　　 PC-9801E/F/Mでは第２水準漢字ROMはオプション
＊5　PC-9801VX/UXのみ
＊6　PC-9801VX/UXのみ実装可．オプション

っています．

第1章は個々のLSIを解説していますが，メモリについては第2章，内部ルーチンについては第3章，CRT等の描画機能関係（VRAM，CRTC，GDCなど）については第4章，フロッピーディスク装置については第5章，また各種インターフェースについては第6章を読んでください．

2.2 BIOS

PC-98システムを構成するLSIには，それぞれに多くのI／O制御命令が用意されているので，多機能であることは確かなのですが，いざ，これらのI／O制御命令を複合化して目的とする動作をさせると，大変な作業になってしまいます．

BIOSは，このデメリットを解消するために用意されている基本ソフトウェアであり，いくつかのコマンドに系統化されているので，ユーザにとって利用しやすいものとなっています．BIOSは，ハードウェアに密着した制御プログラムなので，BASICインタープリタでは提供されていないLSIの隠された潜在能力までも引き出すことができます．BIOSを活用する場合の手続きについては，第3章5を参照してください．

3 CPU(中央処理装置)

CPUは，各種LSIを１つのシステムとしてまとめる，いわばコンピュータの中枢です．このCPUでコンピュータの性能を計ることができます．

PC-9801E/F/Mは，CPUにμPD8086-2*を使用しています．また，PC-9801U/UV/VF/VM/VX/UXでは8086の上位コンパチブルCPUであるV30を使用して性能を向上させています．PC-9801VX/UXではV30に加えi80286***も搭載されており，スイッチにより動作するCPUを切り替えられます．V30はVMとの互換性を高めるために用意されていますが，80286を使用したときにはV30以上の速度が得られます．

3.1 16ビットCPU i8086

i8086はインテル社が開発した16ビットCPUで，処理速度の著しい向上，１Mバイトに及ぶメモリ空間の管理，命令群の充実など，従来の８ビットCPUを大きくしのぐ多くの特長を持っています．しかも，８ビットCPUとの互換性を考慮しているので，８ビットCPUで作成したソフトウェアも一部を変更するだけで利用できます．これまで８ビットCPUに慣れ親しんできたユーザも，さほど抵抗なく16ビットに移行することができます．

図1-2にi8086の端子接続図を示します．i8086は40ピンLSIで，図のように40番ピンがVcc（電源用５Ｖ）の入力端子，１，20番ピンがGND（アース端子）であり，その他の端子にいろいろな信号が割り当てられています．

AD15～AD0(Address and Data)は，アドレスとデータの信号用の端子です．CPUと各種LSIは各々アドレスバス／データバスでつながっていて，そのラインを介してアドレスやデータが出入りします(Ｉ／Ｏ命令，メモリのアクセス)．また，これらの端子はアドレスとデータの２つの信号を１つのラインでやりとりするため，時間的に扱う信号を切り替えて入出力しています(時分割方式)．

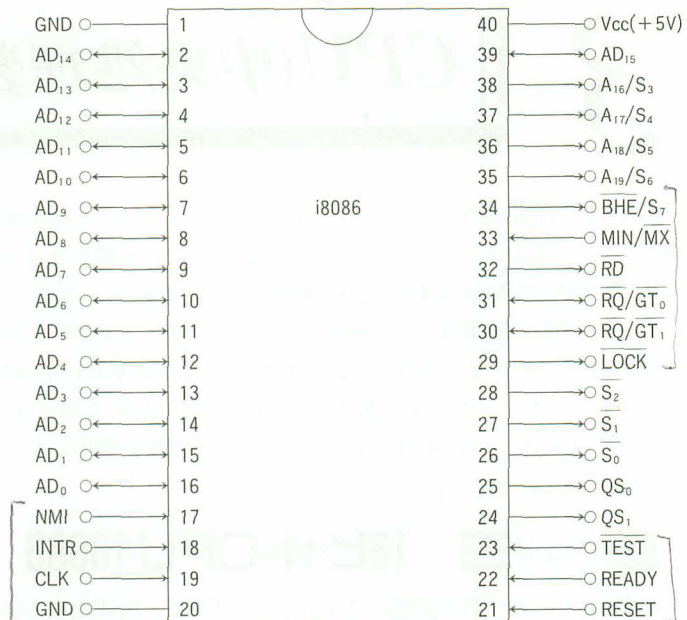
i8086は１Mバイトのメモリ空間を管理するため，アドレスについては20ビットの情報を必要とします．そこで上記AD15～AD0とあわせて，A19～A16がアドレスの信号を出力しています．

---

* 初期製造品にはインテル製i8086が使われている．CPUクロックは5MHz/8MHz切り替え可能．

** 正式名称はμPD70116．U/VF/UXではμPD70116-8(クロック8MHz固定)，VM/UV/VXではμPD70116-10(クロック8MHz，10MHz切替可)を使用．

*** 80286のクロックはVX0/2/4では8MHz固定，VX01/21/41は8MHz/10MHz切替可，UXは10MHz固定．80286については巻末参照．

図1-2
i8086端子接続図

$\overline{S2}$，$\overline{S3}$，$\overline{S0}$ (Status) は，CPUの動作状態（メモリのリード／ライト，I／Oのリード／ライト，命令コードのアクセス，割り込み応答など）を表します．なお，信号名に付いている ̄は，負論理記号の信号であることを示すものです．

READYは，メモリや入力装置などの各LSIからCPUに送られる信号で，入出力の準備ができたことを示します．CPUは，この信号を受けてからデータの入出力を開始するわけです．

RESETは，システムを再スタートさせるための信号です．この信号を受けると，CPUは一度すべての動作を終了させて，新しいアドレスを設定して，実行を再開します．

INTR (Interrupt Request) は，第1章4で解説している割り込みコントローラからCPUに送られる周辺機器の割り込み要求信号です．

CLK (Clock) は，システムがCPUを中心として動作を同期するためのクロック信号で，PC-9801E/F/Mでは8MHz/5MHz，U/VFでは8MHz，VM/UVでは10MHz/8MHzがクロック周波数です．

各々の信号の詳細は，他書に譲ることにします．

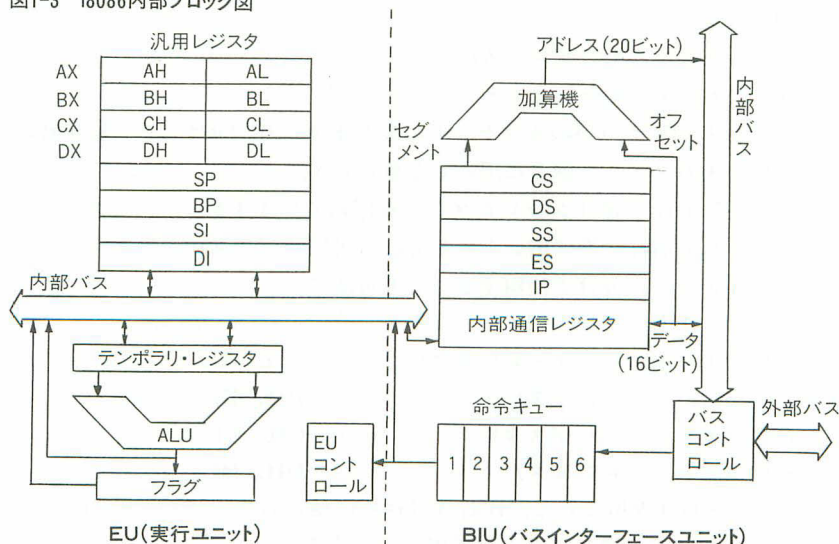
3.2 i8086の内部構造

i8086の大きな特長は，演算処理速度の向上と1Mバイトにも及ぶメモリ空間の直接管理という2点ですが，これらは単に16ビットだからという理由だけではなく，i8086の内部構造に負うところが大きいのです．

図1-3に，i8086の内部ブロック図を示します．

i8086は，EU（実行ユニット）とBIU（バスインターフェースユニット）の2つの部分からなっています．EUは演算処理を行うユニットで，ALU（算術演算ユニット）という部分がレジスタからデータを受けて，演算結果を返します．BIUはアドレスを計算して命令やデータの転送を行うユニットです．

命令キューという部分は，EUが演算などを実行している間に命令のプリフェッチ（先読み）をして，6バイトまで蓄えておくための一時的なメモリです．従来のCPUの場合，演算実行中はバスが遊んでいるし，逆に命令やデータの転送中は演算処理を実行できませんでした．CPUを2つのユニットに分けて，演算処理とバスの管理を各々のユニットに割り当てることによって，バスは常にビジーとなり，演算の実行もデータ転送の終了を待つ必要がなくなるため，処理速度が相対的に向上しています．

図1-3　i8086内部ブロック図

また，i8086は１Mバイトものメモリのアドレスを直接指定するために，セグメントという概念を取り入れています．１Mバイトのアドレスを指定するには20ビット，16進数にして５桁必要ですが，これを16ビットのレジスタを２本使って，上位４桁のアドレスと下位４桁のアドレスを加算することによって表しています．上位４桁をセグメント・アドレスと言い，アドレス指定の基準点となります．下位４桁をオフセット・アドレスと言い，セグメント・アドレスからのずれを示しています（詳しくは第２章に述べてあります）．セグメント・アドレスとオフセット・アドレスの値が図中の加算機に送られて，20ビットのアドレス情報となるわけです．セグメント・アドレスを指定するためには，セグメント・レジスタが用いられ，４本のセグメント・レジスタは各々，

CS（コード・セグメント）
DS（データ・セグメント）
SS（スタック・セグメント）
ES（エクストラ・セグメント）

と呼ばれています．CSはCPUが実行する命令コードが格納されているセグメントを示し，DS，ESはデータ転送時に使います．

オフセット・アドレスの指定には，

IP （インストラクション・ポインタ）
SP （スタック・ポインタ）
BP （ベース・ポインタ）
SI （ソース・インデックス）
DI （デスティネーション・インデックス）

が用いられます．IPは，CSのオフセットになります．SI，DIはデータ転送時の転送元，転送先のアドレス指定に用いると有効です．

SP，BP，SI，DIは，ポインタレジスタと呼ばれていますが，レジスタにはこの他に，主として演算処理に用いられるもの（汎用レジスタ）が４本存在し，上位下位の８ビットに分けて使用することも可能です．

	（８ビット使用時）
AX （アキュムレータ・レジスタ）	………AH，AL
BX （ベース・レジスタ）	………BH，BL
CX （カウンタ・レジスタ）	………CH，CL
DX （データ・レジスタ）	………DH，DL

AXは転送や演算専用として，BX，CXはその他に各々，アドレス間接指定，繰り返し命令などでのカウンタとして用いられます．

3.3 V30の拡張構造

V30は，NECが独自に開発したi8086の上位コンパチブルCPUです．

基本的構造についてはi8086と同じですが，様々な拡張がなされていて，実行速度も向上しています．ここでは，V30の拡張された構造・機能について解説します．

図1-4にV30の内部ブロック図を示します（i8086と比較しやすい形にするため，一部レジスタの名称を変更しています）．

図1-3のi8086の内部ブロック図と比較してみてください．i8086の基本構造にいくつかの拡張を施したCPUであることがわかるでしょう．実際，i8086のソケットにV30を差し替えても正常に動作するようです．では，その拡張された面を具体的に挙げてみましょう．

①サブデータバスの採用

内部を走るデータバスが2系統になっています．このため，2つのデータを同時に転送することが可能となり，実行速度が向上します．例えば，次の演算，

ADD AX, DX

を実行する場合，従来なら3ステップを要するものを2ステップで処理することができます（**表1-1**参照）．

②2本のテンポラリ・レジスタの採用

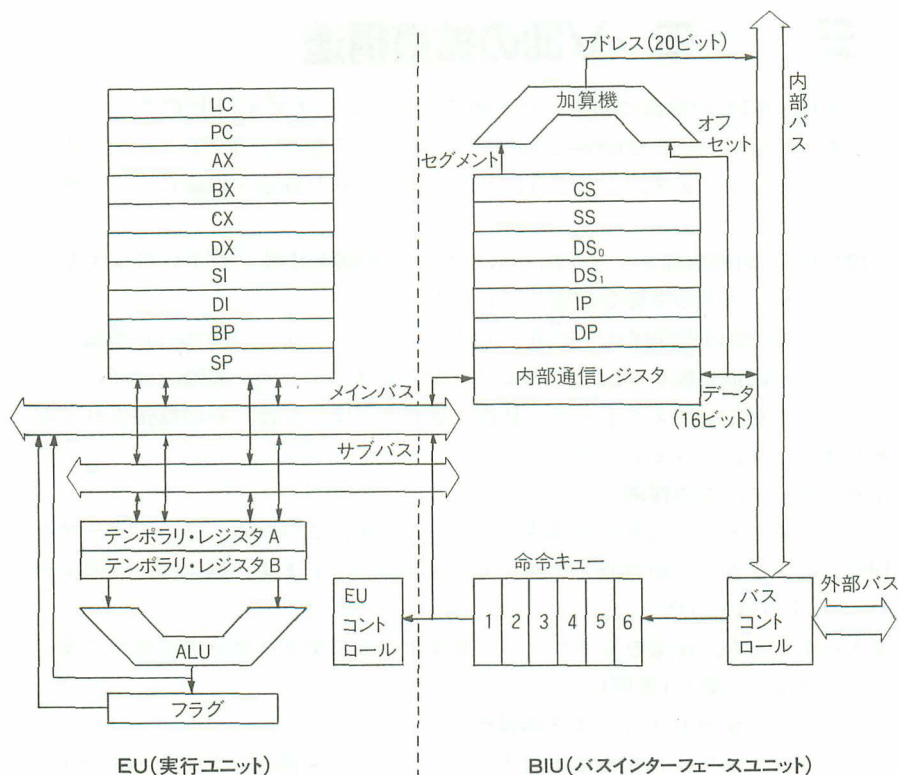
テンポラリ・レジスタを2本にすることによって乗除算，シフト・ローテート命令の高速化が計られています．

③LC（ループカウンタ）の活用

リピート・プリフィックス命令によって制御されるプリミティブ・ブロック転送，および多ビットシフト・ローテート命令を高速化するために，専用レジスタLCを設けています．これによって，上記命令の実行速度は約2倍になります．

表1-1
バスの数によるステップの違い

データバスの数	1（シングルバス）	2（デュアルバス）
ステップ 1	ALU ←— AX	ALU ←— AX, DX
ステップ 2	ALU ←— DX	AX ←— ALU
ステップ 3	AX ←— ALU	——

図1-4　V30内部ブロック図

④スタンバイ機能

プログラムの実行を停止させて，消費電力を大幅に低減する機能で，スタンバイ中の消費電力は実行中の約10分の１となります．プログラムの再開も可能です．

⑤8080エミュレーションモード

Ｖ30には，i8080*の命令をそのまま実行できるエミュレーションモードがあり，Ｖ30の汎用性をより高めています．

以上述べた他に様々な拡張機能がありますが，詳しくは関連書を参照してください．

---

*　i8086を開発する際にベースとなった８ビットＣＰＵ

4 I/Oポート

PC-98では，各種機能を受け持つ周辺LSIがCPUとデータのやり取りをする際，各LSI毎に割り当てられたI/Oポートを介して行われます．

I/Oポートには，I/Oポートアドレス(0000H～FFFFH)が割り付けられており，このアドレスに制御データを出力することでLSIを制御することができます．

ユーザ側からも，各LSIのＩ／Ｏポートアドレスを知っていれば，各LSIを直接制御することが可能となります．BASICレベルでも，Ｉ／Ｏポートを介して各LSIに命令を出すことができ，命令を出すときはOUT命令，情報を得たいときはINP命令を用います（アセンブラでは，IN/OUT命令)．

4.1 I/Oポートアドレス

Ｉ／Ｏポートアドレスの一覧を**表1-2**に示します．アドレスとそれに加えて設定すべきパラメータについての詳細は，Ｉ／Ｏ制御命令を取り扱う各節で解説していますので，ここではその概要だけをまとめておきます．

CPU内部では，ALレジスタを介してＩ／Ｏポートアドレスに制御データを送出することによって，各LSIを制御しています．

なお，**表1-2**のポートアドレスは，例えば15番の５インチ固定ディスクインターフェースの場合，A1，A0に０，１を代入して，

10000000(80H)，10000010(82H)，10000100(84H)，10000110(86H)

の４種類ということです．×印のビットは不定で，通常０としています．

簡単な例を挙げてみます．**表1-2**からシステムポートのＩ／Ｏポートアドレスは31H，33H，35H，37Hの４つであることがわかります．このうち37Hのポート（システムポートのＣポート用）にデータを出力することで，ブザーを鳴らすことができます．

```
MOV AL, 06H
OUT 37H, AL
```

表1-2　I/Oポートアドレス一覧

ポートアドレス																デバイス名	LSI
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
×	×	×	×	×	×	×	×	0	0	0	0	0	×	A0	0	割込コントローラ・マスタ	8259A
×	×	×	×	×	×	×	×	0	0	0	0	1	×	A0	0	割込コントローラ・スレーブ	8259A
×	×	×	×	×	×	×	×	0	0	0	A3	A2	A1	A0	1	DMAコントローラ	8237A
×	×	×	×	×	×	×	×	0	0	1	0	×	×	×	0	カレンダ時計*1	1990A
×	×	×	×	×	×	×	×	0	0	1	0	×	A1	A0	1	DMAバンク	
×	×	×	×	×	×	×	×	0	0	1	1	×	×	A0	0	RS-232CI/F	8251A
×	×	×	×	×	×	×	×	0	0	1	1	×	A1	A0	1	システムポート	8255A
×	×	×	×	×	×	×	×	0	1	0	0	×	A1	A0	0	プリンタI/F	8255A
×	×	×	×	×	×	×	×	0	1	0	0	×	×	A0	1	キーボードI/F	8251A
×	×	×	×	×	×	×	×	0	1	0	1	×	×	A0	0	NMI	
×	×	×	×	×	×	×	×	0	1	0	1	×	A1	A0	1	320KBFDI/F*2	8255A
×	×	×	×	×	×	×	×	0	1	1	0	A2	A1	A0	0	CRTコントローラ・テキスト	7220A
×	×	×	×	×	×	×	×	0	1	1	0	×	×	×	1	予約	
×	×	×	×	×	×	×	×	0	1	1	1	A2	A1	A0	0	CRTコントローラ	52611
×	×	×	×	×	×	×	×	0	1	1	1	×	A1	A0	1	タイマコントローラ	8253
×	×	×	×	×	×	×	×	1	0	0	0	0	A1	A0	0	固定ディスクI/F*3	
×	×	×	×	×	×	×	×	1	0	0	0	0	A1	A0	1	BRANCH4670	
×	×	×	×	0	0	0	1	1	0	0	0	1	A1	A0	0	サウンドボード*4	YM2203
×	×	×	×	×	×	×	×	1	0	0	0	1	A1	A0	1	ネットワークI/F*5	
×	×	×	×	×	×	×	×	1	0	0	1	×	A1	A0	0	1MBFDC*6	765A
×	×	×	×	×	×	×	×	1	0	0	1	0	A1	A0	1	CMTI/F*5	8251A
×	×	×	×	×	×	×	×	1	0	0	1	1	0	A0	1	GP-IBスイッチ*5	
×	×	×	×	×	×	×	×	1	0	0	1	1	1	0	1	予約	
×	×	×	×	×	×	×	×	1	0	1	1	A0	A1	A0	0	CRTコントローラ・グラフ	7220A
×	×	×	×	×	×	×	×	1	0	1	0	A2	A1	A0	1	文字パターンROM	
×	×	×	×	×	×	×	×	1	0	1	1	A0	A1	A0	0	通信制御アダプタ	7201
×	×	×	×	×	×	×	×	1	0	1	1	0	A1	A0	1	〃	8255A
×	×	×	×	×	×	×	×	1	0	1	1	1	A1	A0	1	〃	8253A
×	×	×	×	×	×	×	×	1	0	1	1	1	×	×	0	予約	
×	×	×	×	×	×	×	×	1	0	1	1	A3	A2	A1	A0	RS-232C拡張I/F	
×	×	×	×	×	×	×	×	1	0	1	1	1	1	1	0	1MB/640KB切換I/O*7	
×	×	×	×	×	×	×	×	1	1	0	0	1	A1	A0	0	640KBFDC*8	765A
×	×	×	×	×	×	×	×	1	1	0	0	A2	A1	A0	1	GP-IB*5	7210
×	×	×	×	×	×	×	×	1	1	0	1	×	×	×	×	未使用（ユーザ使用可）	
1	0	1	1	1	1	1	1	1	1	0	1	1	0	0	0	予約*10	
1	0	1	1	1	1	1	1	1	1	0	1	1	0	1	0	予約*10	
1	0	1	1	1	1	1	1	1	1	0	1	1	0	1	1	マウス割込周期設定*10	
0	0	1	1	1	1	1	1	1	1	0	1	1	0	1	1	内部サウンド周波数設定*10	8253
0	0	1	1	1	1	1	1	1	1	0	1	1	1	1	1	I/Oアドレス77Hと同じ*10	8253
0	1	1	1	1	1	1	1	1	1	0	1	1	A1	A0	1	マウスコントロール*10	8255A
×	×	×	×	×	×	×	×	1	1	1	0	0	0	0	0	キーボード(スキャン方式)*9	
								|								|	
×	×	×	×	×	×	×	×	1	1	1	0	1	0	1	1	〃	
×	×	×	×	×	×	×	×	1	1	1	0	1	1	×	×	未使用（ユーザ使用可）	
×	×	×	×	×	×	×	×	1	1	1	1	0	×	×	1	未使用（ユーザ使用可）	

＊1　VM21/UV21はμPD4990A
＊2　E/F/Mのみ内蔵
＊3　F3/M3/VM4は内蔵，他オプション
＊4　UV2, 21は内蔵，他オプション
＊5　オプション
＊6　M/VM/UVは内蔵，他オプション
＊7　VM/UVのみ内蔵
＊8　F/U/VF/VM/UVは内蔵，他オプション
＊9　BASICのINP関数でのみ有効(機械語レベルではユーザ使用可)
＊10　U/VF/VM/UVのみ

このようにI/Oポートの37Hに06Hを出力するとブザーが鳴ります．停止したいときは，

```
MOV   AL, 07H
OUT   37H, AL
```

のように07Hを出力してやればよいのです．

I/O制御命令は，デバイスに連続してアクセスする場合にI/Oデバイスのタイミングを考慮する必要かあったり，いくつものI/O制御命令を組み合わせなければならなかったりするので，BIOSを利用したほうが簡単にプログラムを組むことができます．また，それ以外の理由からも，I/O制御命令を使うよりBIOSを利用する方が望ましいと言えます．

これは，前に挙げたブザーを鳴らす場合のように，I/O制御命令を使っても簡単に操作できる場合にも言えることです．なぜなら，I/Oを直接アクセスしてしまうと，将来ハードウェアの変更があった場合に，そのプログラムは正常に動作しなくなってしまうからです．しかし，BIOSを利用していれば，ROM内のBIOSはハードウェアに対応しているため，ハードウェアの違いはBIOSによって吸収させることができます．

プログラムを作るときには，BIOSに用意されている機能ならばI/O制御をしないで積極的にBIOSを使うべきです．

4.2 I/O制御命令アクセス時の制限

マシン語でプログラムを作成する際，連続して同じLSIに対してＩ／Ｏ制御命令を出すと，正常に動作しない場合があります．これは，前回の命令に対する処理が終了しないうちに新しい命令を出した場合に起こる現象です．こうした誤動作を防ぐために，通常アクセスの際にはある程度のNOP（ノーオペレーション）時間を設けて，処理終了を待つ必要があります．LSIによってその時間は異なりますが，**表1-3**に示す値を目安にすれば無難といえます．

表に示した数のNOPをI/Oアクセス命令間に入れれば，PC-9801E/F/M/U/VF/VM/UV/VX/UX(VX/UXはV30動作時)のどの機種，どのクロック状態でも十分なウェイトが得られます．

表1-3　I/O命令連続アクセス時のNOP挿入数一覧

周辺LSI名	IN→IN	OUT→OUT	OUT→IN	IN→OUT
8255A-5　PPI	1	1	3	1
8253-5　タイマ	2	2	3	1
8251A USART　モード初期化	0	6	0	0
8251A USART　ライトデータ同期	0	20	0	0
8251A USART　ライトデータ非同期	0	9	0	0
8259A　PIC	0	0	1	0
765AC　FDC	0	0	0	0
7220A　GDC（グラフ）*	2	2	3	2
7220A　GDC（テキスト）	0	0	1	0
8237-5　DMAC	0	0	1	0
7210　GP-IB	0	0	1	0

＊　GDCクロック2.5MHz，高解像度CRT，スーパーインポーズしない場合

5 割り込み(Interrupt)

普段の生活で「割り込み」という言葉はあまりよい意味で用いられませんが，コンピュータのハードウェアを取り扱うことには，この「割り込み」をいかに活用するかによって，そのシステムの性能を左右するくらい重要な技術になります．PC-98はその複雑なシステムを効率よくコントロールするために，数多くの割り込みを使用しています．

5.1 割り込みとは

コンピュータの「割り込み」は，日常生活にあてはめるとよく理解することができます．例えば，仕事をしているときに電話が鳴ると，仕事の手を休めて電話に出て，それが終ったらもとの仕事に戻りますが，この場合の電話が割り込みに相当します．

ここで重要なのは，電話がかかるとそれを人に知らせるためにベルが鳴る，ということです．もし，電話がかかってきたときに小さなランプが点灯するだけだったとしたら，電話を取るために一日中電話を監視しているか，仕事をするために電話を無視するかのどちらかしかありません．つまり，割り込みとはそれ（電話）を監視することなしに，必要なとき（電話がかかってきた時）一時的に別の仕事（電話をとること）をさせる行為をいうのです．

この話をそのままPC-98にあてはめてみましょう．PC-98は，キー入力待ちになる前にキーを押しても（先行入力という）押されたキーを覚えていますが，これを例にとって説明します．

いまCPUはある計算をひたすら実行しているところだとします．もちろん，このときボードの監視などはしていません．ところが，キーが押されるとハードウェアから割り込みが発生するので，CPUはいま実行している計算をひとまず中断して，キー入力の処理をするためにサブルーチンジャンプします．それが終わるとリターンして中断していた計算を続行します．

キーの入力があるとCPUに割り込みがかかるので，キーボードを監視していなくても押されたキーを常にキャッチすることができますが，これは前の電話の話にそっくり当てはまります．

つまり，CPUの割り込みとは，ハードウェアからの合図でサブルーチンジャンプする機能ということになります．

BASICにも割り込み機能があり，ON KEY GOSUB, KEY ON等の命令を使うと特定のキー（ファンクションキーなど）が押されたときにサブルーチンにジャンプすることができるようになっています．（ただしこれはCPUレベルの割り込みで実現しているわけではありませんが）．CPUの割り込みもイメージとしてはBASICの割り込みと同じように考えることができます．

なお，割り込みとCPUの動作とは密接な関係があります．PC-98は，機種によって8086，V30，80286の３種類のCPUを使用していますが，初代の9801は8086を使用していたこともあって，8086を基準に設計されています．このため，本書でも8086の場合について解説することにします（V30，802306の場合もほとんど同じ）．

5.2 8086の割り込み

8086のCPUには，**表1-5**のように大きく分けて２種類の割り込みがあり，先に説明した割り込みは，表のうち①，②の「外部割り込み」に相当します．

８ビットCPUでは割り込みというと主に「外部割り込み」について指していましたが，16ビットCPUである8086は８ビットCPUのそれよりも進化した設計になっているため，「内部割り込み」という概念が加わりました（８ビットCPUにも内部割り込みはありますが，16ビットCPUほど完全なものではありません）．

8086は外部割り込み，内部割り込みあわせて256種類のサブルーチンを取り扱うとこができます．256種の割り込みには０～FFH（255）の番号をつけて区別していて，それぞれのサブルーチンの処理アドレスは任意に設定することができます．**表1-5**にはINTnという表記がいくつかありますが，nの部分の数値が割り込み番号を表しているわけです．

この256種類の割り込み処理ルーチンのアドレスは，ベクタテーブルと呼ばれる0000：0000H～0000：0400Hの１バイトのエリアに前もって準備しておきます（１つの割り込み番号に対し，セグメント＋オフセットの４バイト×256種類＝１Kバイト）．これは，BASICのキー割り込みのON KEY GOSUBで，あらかじめ割り込みのサブルーチンを指定しておくのと似ています．

外部割り込みのことを「ハードウェア割り込み」，内部割り込みを「ソフトウェア割り込み」ということもある．
また，割り込みを英語では「インタラプト（Interrupt）」と言う．

割り込みが発生すると，CPUはベクタテーブルを参照して割り込み番号に対応したアドレスに処理を移します．例として**表1-4**に9801のベクタテーブルの先頭部分をダンプしたものを示します．これを見ると，たとえばINT　0の処理ルーチンは，セグメント0646H，オフセット2C0CHにあることがわかります．

割り込み処理ルーチンからリターンするには，IRET (Interrupt Return) 命令を用います．

表1-4　PC9801のベクタテーブルの例

(INT0～7のベクタアトレスを示す)

```
0000：0000  0C  2C  46  06  2B  08  80  FD-E5  07  80  FD  2B  08  80  FD
0000：0010  2B  08  80  FD  CE  51  60  00-F8  50  60  00  4C  02  45  28
```

INT 0のエントリは0646：2C0C，INT 1のエントリはFD80：082B，……

表1-5　8086ＣＰＵの割り込みの分類

割り込み {
- 外部割り込み {
  - NMI ……………………………①INT　2
  - INTR ……………………………②INT　8～17H(9801の場合) (Interrupt Request.)
- 内部割り込み {
  - CPU自身により呼び出し …………③INT　0，1，3，4
  - ユーザによる呼び出し……………④上記以外のINT(0FFHまで)

		割り込みの発生原因	備　考	優先順位
外部	①	NMI入力の立ち上がりで発生	禁止不可能	2
	②	INTR入力がハイレベルになると発生	禁止可能	3
内部	③	INT　0―ゼロで除算すると発生	禁止不可能	1
		INT　1―シングルステップ	デバック用	4
		INT　3―ブレークポイント	デバッグ用	1
		INT　4―オーバーフロー	禁止可能	1
	④	プログラム中のINT命令により発生	禁止不可能	1

○V30のINT 5 は，正確には③に分類されるが，9801では④として使っているのでここでも④に含めた．

○80286のINT　10Hは③に分類される．また，80826のINT 5 ～ 9，0D0Hは，正確には③に分類されるが，9801では②または④として使っている．

○NMI (Non Maskable Interrupt)という端子がCPUにある．

○INTR (Interrupt)という端子がCPUにある．これは割り込みコントローラで15本に拡張される．

○INT (Interrupt)という命令がCPUにある．

表1-5では，外部割り込みをMNIとINTRに分類していますが，このうちのINTR割り込み（マスク可能割り込み）は，割り込みコントローラというLSI(8259A　PIC＝Programmable Interrupt Controller）を使って複数の外部割り込みを受け付けられるようにしています．9801では２個のPICをカスケードと接続して，入力を15本に拡張しています（２個のPICはマスタ/スレーブと呼び分ける）．

PICの役割は，割り込み入力本数の拡張，各々の割り込み入力の禁止/許可，割り込み優先順位の処理，CPUへの割り込み番号の出力などです．最後の「割り込み番号の出力」という機能のために，外部割り込みも内部割り込みとほとんど区別しないで取り扱えるのです．

ハードウェア的な面で注意すべき点は，**表1-5**の「INTR入力がハイレベルになると発生」という部分です．この記述はCPUのINTR端子のことを言っていて，PICの入力端子のことではありません．9801ではPICの初期化の際に，「PICの割り込み入力端子の立ち上がりを割り込み要求とする」ように設定しています．外部のハードウェアから割り込みをかけるときには，信号が立ち上がらなければ（ハイレベルのままでは）割り込みが起こったと認識されないので注意してください．

外部割り込みが，ハードウェアからの割り込み要求でサブルーチンジャンプするのに対し，内部割り込みはハードウェアからではなく，ソフトウェアからの割り込み要求でサブルーチンジャンプする機能です．もっと平易に言い替えれば，内部割り込みはサブルーチンジャンプの一種といってもよいでしょう．

5.3 内部割り込みの利用

外部割り込みの機能を使うことで，効率よくハードウェアをコントロールすることができることはわかりましたが，内部割り込み（INT）にはどのような価値があるのでしょう．

機能的にはCALLとほぼ同じですが，INTとCALLの一番の違いは，サブルーチンの呼び出し方法であるといえます．

CALL命令は，サブルーチンにジャンプするためにその開始アドレスを指定しますが，INT命令は，割り込み番号を介してサブルーチンを指定します．

たとえば，あるサブルーチンがF000：0000Hというアドレスから始まっているとして，CALL命令でジャンプするときと，INT命令でジャンプするときとを比べてみると，

○CALLでは,

　　CALL　F000：0000H

のようにアドレスを直接指定して実行します.

○INTでは,まず何番の割り込み番号を使うかを決めます.ここでは仮に40Hだとすると,0000：0100H～0103H番地にF000：0000Hというアドレスをセットしておいてから,

　　INT　40H

を実行します.つまり,サブルーチンのアドレスを間接的に指定しているわけです.

INT命令の利点とは,ずばりサブルーチンのアドレスが変わったとしてもプログラムを書き換える必要がないことにあります.

PC-98のROMの中にはBIOSと呼ばれるサブルーチンが入っていますが,そのサブルーチンのアドレスはPC-98のバージョンによって一定ではありません.しかし,それらのサブルーチンはINT命令で呼び出すことができるように電源ON時にPC-98が自動的にセットしてくれるため,ユーザがBIOSをコールするときにはBIOSに与えられている割り込み番号で呼び出すことができます.

他にINT命令がCALLと違うところは,サブルーチンにジャンプするときに戻り番地だけでなく,フラグもスタックに保存されることです.つまりスタックは3×2＝6バイト消費します.また,INTでサブルーチンにジャンプすると,自動的にCPUの割り込み許可フラグがクリアされます(CPU命令のCLIの実行と評価).サブルーチン内で割り込みを禁止する必要がない場合には,STI(割り込み許可)命令を実行して割り込み許可状態にしておきます.

PC-98では,INT　18H～1CHがBIOS用に予約されています.各BIOSの内容については第1章6以降を参照してください.BIOSのなかには直接I/O命令で書いてもそれほど複雑でない処理がありますが,BIOSでできる処理であれば,できるだけBIOSを利用するべきです.これは,プログラムの互換性を高めるためには重要なことです.

たとえば,PC-98の時計用LSIは,VX,VM21,UV21と,それ以前の機種とではLSIの種類が違うので,時計用LSIに直接I/O命令でアクセスするとVX等とそれ以前の機種とで共用できるプログラムを作ることは面倒になります.しかし,BIOSを用いて時計にアクセスしていれば,LSIが違っていても,時計とのやりとりはBIOSがやってくれるので,ユーザはLSIの種類を意識することなくプログラムを作ることができるのです.

5.4 外部割り込みの利用

外部割り込みは，内部割り込みとほぼ同じように利用できますが，割り込みコントローラ(PIC)にいくつか命令を与えるという手順が増えます．5.2でPICの役割を説明しましたが，そのうちユーザがプログラムを作るときに必要な操作は，「各々の割り込み入力の禁止/許可」，割り込み優先順位の処理」についてです．

外部割り込みが受け付けられるようにするには，CPUの割り込み許可フラグをセット（CPU命令STIを実行）します．さらに，該当する外部割り込みを許可するために，PICの割り込みマスクレジスタを操作します．

外部割り込みの処理ルーチンは，処理の最後にPICに対して割り込み処理が終了したことを通知します．これは割り込み優先順位の処理に当たります。

CPUが割り込み処理ルーチンを実行すると，PICはその割り込みより優先順位の低い割り込みをすべて割り込み禁止にします．このため，割り込みルーチンの最後で処理が終了したことをPICに知らせないと，それより低い割り込みがいつまでも受け付けられなくなってしまうからです．

リスト1-1

```
○割込マスクレジスタ（ＩＭＲ＝Interrupt Mask Register）の操作
・INT 08H～0EHを許可／禁止する場合（マスタＰＩＣを操作）
        PUSH  AX
        IN    AL,02H  ;Read IMR(master)
        AND   AL,FEH  ;INT 08H 許可(該当ビットのクリアで割込許可)
        OUT   AL,02H  ;Write IMR(master)
        POP   AX
・INT 10H～1FHを許可／禁止する場合（スレーブＰＩＣを操作）
        PUSH  AX
        IN    AL,0AH  ;Read IMR(slave)
        OR    AL,04H  ;INT 12H 禁止(該当ビットのセットで割込禁止)
        OUT   AL,0AH  ;Write IMR(slave)
        POP   AX

○割込処理終了の通知（マスタ・スレーブ兼用）
        PUSH    AX
        MOV     AL,0B    ;ISR Read mode(master)
        OUT     00,AL
        NOP
        IN      AL,00    ;Read ISR(master)
        TEST    AL,80    ;IR7(Slave PIC)がサービス中なら
        JNZ     yyy      ;yyyにジャンプ
xxx:    MOV     AL,20    ;マスタにEOI(End Of Interrupt)を送る
        OUT     00,AL
        POP     AX
        JMP     zzz
yyy:    MOV     AL,20    ;スレーブにEOIを送る
        OUT     08,AL
        MOV     AL,0B    ;ISR Read mode(slave)
        OUT     08,AL
        NOP
        IN      AL,08    ;Read ISR(slave)
        OR      AL,AL    ;スレーブＰＩＣにサービス中の割込が無くなれば
        JZ      xxx      ;xxxにジャンプ
        POP     AX
zzz:
```

5.5 PC-98の割り込み一覧

PC-98では**表1-6**に示すように割り込みを使用しています.

各割り込みは次のように分類することができます.

○INT 00, 01, 03, 04HはCPUが自ら発生する割り込みです.

○INT 2,08〜17Hは, 各ハードウェアからの割り込み要求によって発生します. 優先順位はINT 08Hが最も高く, INT 17Hが最低です.

○INT 18〜1CHはBIOSです. BIOSはBASIC動作時だけでなく, MS-DOS, CP/M動作時にも利用することができます.

○INT 05, 06, 07Hはそれぞれ,「COPYキーが押された時」,「STOPキーが押された時」,「インターバルタイマで設定した時間が経過した時」にBIOSから呼び出されます.

○INT 40〜7FHはユーザに解放されています.

表1-6-(1) PC-9801割り込みベクタ用途一覧

ベクタ[*1] 番号	ベクタ[*1] アドレス	種別	用途(割込名)
00	00-03	CPU予約	除算エラー
01	04-07	CPU予約	シングルステップ
02	08-0B	CPU予約	NMI (外部割込)
03	0C-0F	CPU予約	ブレークポイント
04	10-13	CPU予約	オーバーフロー
05	14-17		COPYキー
06	18-1B		STOPキー
07	1C-1F		インターバルタイマ
08	20-23	外部割込IR0	タイマ (8253)
09	24-27	外部割込IR1	キーボード (8251A)
0A	28-2B	外部割込IR2	CRTV (マスタμPD7220 V-SYNC)
0B	2C-2F	外部割込IR3	拡張スロットINT0
0C	30-33	外部割込IR4	RS-232C (8251A)
0D	34-37	外部割込IR5	拡張スロットINT1 (CMT)
0E	38-3B	外部割込IR6	拡張スロットNT2 (ODAプリンタ)
0F	3C-3F	外部割込IR7	スレーブPIC (8259A)
10	40-43	外部割込IR8	セントプリンタ (8255A *2
11	44-47	外部割込IR8	拡張スロットINT3 (ハードディスク)
12	48-4B	外部割込IR10	拡張スロットINT41 (640KBFD)
13	4C-4F	外部割込IR11	拡張スロットINT42 (1MBFD)
14	50-53	外部割込IR12	拡張スロットINT5
15	54-57	外部割込IR13	拡張スロットINT6 (マウス)
16	58-5B	外部割込IR14	NDP (8087) *3
17	5C-5F	外部割込IR15	ノイズ(GND)

表1-6-(2) PC-9801割り込みベクタ用途一覧

ベクタ*1		種 別	用途（割込名）	
番号	アドレス			
18	60-63	BIOS	キーボード，CRT	*4
19	64-67	BIOS	RS-232C	
1A	68-6B	BIOS	カセット，プリンタ	
1B	6C-6F	BIOS	ディスク	
1C	70-73	BIOS	カレンダ，インターバルタイマ	
1D	74-77	システム予約		
1E	78-7B		N88BASIC	
1F	7C-7F	システム予約		
20	80-83	MS-DOS	プログラムの終了	*5
21	84-87	MS-DOS	システムコール	
22	88-8B	MS-DOS	プログラム終了アドレス	
23	8C-8F	MS-DOS	CTRL-Cアドレス	
24	90-93	MS-DOS	致命的エラー処理アドレス	
25	94-97	MS-DOS	物理セクタ読み出し	
26	98-9B	MS-DOS	物理セクタ書き込み	
27	9C-9F	MS-DOS	プログラムの常駐終了	
28 | 3F	A0-A3 FC-FF	システム予約		*6
40 | 7F	100-103 10C-1FF	ユーザ用	ユーザが自由に使うことができる	
80 | FF	200-203 3FC-3FF	システム予約		

*1 ベクタアドレスのセグメントは0000H
*2 VXの80286モードでは，NDPの割り込みになる
*3 VXの80286モードでは，未使用になる
*4 INT 18H～1CHに割り当てられているBIOSは次の箇所で解説している
- ・キーボードBIOS…………(第1章6.3)
- ・CRT…………………………(第4章6)
- ・RS-232C……………………(第6章2.4)
- ・プリンタ…………………(第6章5.3)
- ・ディスク…………………(第5章3)
- ・カレンダ…………………(第1章8.3)
- ・インターバルタイマ……(第1章7.3)

*5 MS-DOS固有の機能なので詳しくは関連書に譲る．
*6 INT 33Hは一般的にマウスドライバで使われる．

6 キーボード

PC-98システムは，CPUを中心に各種周辺装置を制御するための専用インターフェースを備えていて，様々な機能を実現しているわけですが，システムとユーザの間のインターフェースも当然必要です．

ユーザがシステムに命令を入力するためのインターフェースとなるのがキーボードで，PC-98では，キーボード自体がマイクロプロセッサを搭載して1つのユニットを形成しています．ここでは，システムとキーボードの間におけるデータ処理の方式を解説して，キー入力に関する基本的な制御法を示します．

6.1 キーボードインターフェース

ユーザはキーボードからコードを入力しますが，そのデータはカール・コードを介してシステム本体に送られます．キー操作した内容はどのようにしてCPUに伝わるのでしょうか．まず，キーボード周辺のブロック図を**図1-9**に示します．

キーボードのキーを操作すると，その時点でのキー状態に対応するデータが，キーマトリックスから選択されます．キーボード内のマイクロプロセッサμPD8048Aは，このデータをカールコードを介して本体側のキーボードインターフェースμPD8251Aに，シリアルデータとして出力します．このシリアルデータ形式は，

図1-9　キーボード周辺ブロック図

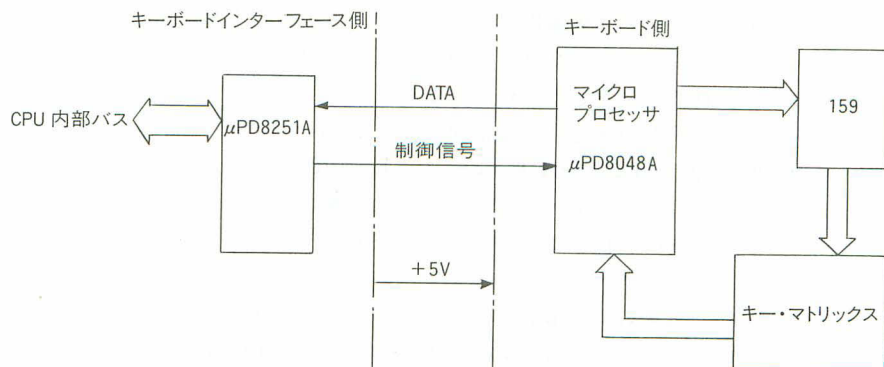



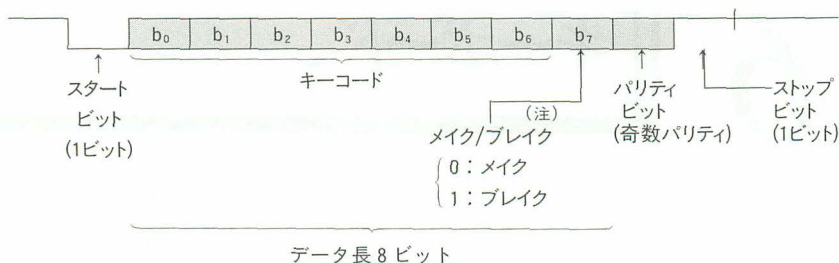
図1-10 シリアルデータ形式

（注）メイク：キーが押されたときの割り込みを示す．
ブレイク：キーが離されたときの割り込みを示す．

図1-10のようになっています．

キーコードは，キーボード上の各キーに対応するコードで，**表1－6(1)**のように割り当てられています．キーボードからは，キーが押されたときと離されたときにデータが出力されます．1つのキーを押して離すと，キーコードは同じで，最上位ビットだけ異なるデータが出力されることになります．例えば，キーコードが70Hであるシフトキーを押す（メイク状態）と70Hというデータ，離す（ブレイク状態）とF0Hというデータが発生するわけです．このように，キーのメイク／ブレイク状態を示す1ビットのフラグに7ビットで表現されるキーコードを追加して，8ビットのデータを構成したものをスキャンコードと呼びます．スキャンコードとキーコードとの関係は，次式のようになります．

nnH＝mmH　　　　　メイク時(キーが押された状態)
nnH＝mmH＋80H　　ブレイク時(キーが離された状態)

ANKキー（英数字カナキー）は，1つのキーに複数のキャラクタが対応しているので，シフトキー（SHIFT，CAPS，カナ，GRPH，CTRL）との組み合わせで入力するキャラクタを選択しています．ですから，1つのキーコードだけからは，ユーザがどのキャラクタを指定しているのか判別できず，シフトキーが同時に押されているかどうかが問題となります．このように，キーコードとシフトキーの状態とで決定される情報に対して，1バイトの内部コード(キーコード表参照)を割り当てています．キーコードと内部コードからなる2バイトデータのことを，キーコードデータと呼びます（**表1－6(2)**参照）．

キーコードデータの上位1バイトがキーコード，下位1バイトが内部コードです．

キーボードから送られてくるシリアルデータは，キーボードインターフェースμPD8251Aによってバイト単位のパラレルデータに変換されます．

表1-6(1)キーコード

上位3ビット→ / 下位4ビット↓

	0	1	2	3	4	5	6	7
0	ESC	Qタ	Fハ	,<ネ	−	.	STOP	SHFT
1	1!ヌ	Wテ	Gキ	.>ル	／	NFER	COPY	CAPS
2	2"フ	Eイ	Hク	/?メ	7		f･1	カナ
3	3#ア	Rス	Jマ	_ロ	8		f･2	GRPH
4	4$ウ	Tカ	Kノ	空白	9		f･3	CTRL
5	5%エ	Yン	Lリ	XFER	＊		f･4	
6	6&オ	Uナ	;＋レ	RLUP	4		f･5	
7	7'ヤ	Iニ	:＊ケ	RLDN	5		f･6	
8	8(ユ	Oラ	[{ム	INS	6		f･7	
9	9)ヨ	Pセ	Zツ	DEL	＋		f･8	
A	0ワ	@~゛	Xサ	↑	1		f･9	
B	-ニホ	[{゜	Cソ	←	2		f･10	
C	^ヘ	復改	Vヒ	→	3			
D	¥|ー	Aチ	Bコ	↓	＝			
E	BS	Sト	Nミ	HMCR	0			
F	TAB	Dシ	Mモ	HELP	,			

キーボード上のキーひとつひとつにつけられた通し番号がキーコード.
例えば,［G］キーのキーコードは21H.
復改はリターンキー,空白はスペースキー.
フルキーの上段に並んでいる数字キーとテンキー上の数字キーとは区別できるが,2個のリターンキー,シフトキーは区別できない.

表1-6(2) キーコードデータ

キーコード		通常		SHIFT		CAPS		CAPS+SHIFT		カナ		カナ+SHIFT		GRPH		CTRL	
00	ESC	00	1B	00	1B	00	1B	00	1B	00	1B	00	1B	00	1B		
01	! 1 ヌ	01	31	01	21	01	31	01	21	01	C7	01	C7				
02	" 2 フ	02	32	02	22	02	32	02	22	02	CC	02	CC				
03	# 3 ア ァ	03	33	03	23	03	33	03	23	03	B1	03	A7				
04	$ 4 ウ ゥ	04	34	04	24	04	34	04	24	04	B3	04	A9				
05	% 5 エ ェ	05	35	05	25	05	35	05	25	05	B4	05	AA	05	F2		
06	& 6 オ ォ	06	36	06	26	06	36	06	26	06	B5	06	AB	06	F3		
07	' 7 ヤ ャ	07	37	07	27	07	37	07	27	07	D4	07	AC	07	F4		
08	(8 ユ ュ	08	38	08	28	08	38	08	28	08	D5	08	AD	08	F5		
09	) 9 ヨ ョ	09	39	09	29	09	39	09	29	09	D6	09	AE	09	F6		
0A	0 ヲ ワ	0A	40	0A	20	0A	30	0A	20	0A	DC	0A	A6	0A	F7		
0B	= - ホ	0B	2D	0B	3D	0B	2D	0B	3D	0B	CE	0B	CE	0B	8C		
0C	∧ へ	0C	5E	0C	5E	0C	5E	0C	5E	0C	CD	0C	CD	0C	8B	0C	1E
0D	¦ ¥ ー	0D	5C	0D	7C	0D	5C	0D	7C	0D	B0	0D	B0	0D	F1	0D	1C
0E	BS	0E	08	0E	08	0E	08	0E	08	0E	08	0E	08	0E	08	0E	08
0F	TAB	0F	09	0F	09	0F	09	0F	09	0F	09	0F	09	0F	09	0F	09

キーコード		通常		SHIFT		CAPS		CAPS+SHIFT		カナ		カナ+SHIFT		GRPH		CTRL	
10	Q タ	10	71	10	51	10	51	10	71	10	C0	10	C0	10	9C	10	11
11	W テ	11	77	11	57	11	57	11	77	11	C3	11	C3	11	9D	11	17
12	E イ ィ	12	65	12	45	12	45	12	65	12	B2	12	B2	12	E4	12	05
13	R ス	13	72	13	52	13	52	13	72	13	BD	13	BD	13	E5	13	12
14	T カ	14	74	14	54	14	54	14	74	14	B6	14	B6	14	EE	14	14
15	Y ン	15	79	15	59	15	59	15	79	15	DD	15	DD	15	EF	15	19
16	U ナ	16	75	16	55	16	55	16	75	16	C5	16	C5	16	F0	16	15
17	I ニ	17	69	17	49	17	49	17	69	17	C6	17	C6	17	E8	17	09
18	O ラ	18	6F	18	4F	18	4F	18	6F	18	C7	18	C7	18	E9	18	0F
19	P セ	19	70	19	50	19	50	19	70	19	BE	19	BE	19	8D	19	10
1A	~ @ ゛	1A	40	1A	7E	1A	7E	1A	40	1A	DE	1A	DE	1A	8A	1A	00
1B	{ [「 ゜	1B	5B	1B	7B	1B	7B	1B	5B	1B	DF	1B	DF			1B	1B
1C	↵	1C	0D	1C	0D	1C	0D	1C	0D	1C	0D	1C	0D	1C	0D	1C	0D
1D	A チ	1D	61	1D	41	1D	41	1D	61	1D	C1	1D	C1	1D	9E	1D	01
1E	S ト	1E	73	1E	53	1E	53	1E	73	1E	C4	1E	C4	1E	9F	1E	13
1F	D シ	1F	64	1F	44	1F	44	1F	64	1F	BC	1F	BC	1F	E6	1F	04

(注) 表中の2バイトデータがキーコードデータ．このうち，上位1バイトがキーコード，下位1バイトが内部コード．

キーコード		通常		SHIFT		CAPS		CAPS + SHIFT		カナ		カナ + SHIFT		GRPH		CTRL	
20	F ハ	20	66	20	46	20	46	20	66	20	CA	20	CA	20	E7	20	06
21	G キ	21	67	21	47	21	47	21	67	21	B7	21	B7	21	EC	21	07
22	H ク	22	68	22	48	22	48	22	68	22	B8	22	B8	22	ED	22	08
23	J マ	23	6A	23	4A	23	4A	23	6A	23	CF	23	CF	23	EA	23	0A
24	K ノ	24	6B	24	4B	24	4B	24	6B	24	C9	24	C9	24	EB	24	0B
25	L リ	25	6C	25	4C	25	4C	25	6C	25	D8	25	D8	25	8E	25	0C
26	; + レ	26	3B	26	2B	26	2B	26	2B	26	DA	26	DA	26	89		
27	: * ケ	27	3A	27	2A	27	2A	27	2A	27	B9	27	B9	27	94		
28	}] 」 ム	28	5D	28	7D	28	7D	28	7D	28	D1	28	A3			28	1D
29	Z ッ ツ	29	7A	29	5A	29	5A	29	7A	29	C2	29	AF	29	80	29	1A
2A	X サ	2A	78	2A	58	2A	58	2A	78	2A	BB	2A	BB	2A	81	2A	18
2B	C ソ	2B	63	2B	43	2B	43	2B	63	2B	BF	2B	BF	2B	82	2B	03
2C	V ヒ	2C	76	2C	56	2C	56	2C	76	2C	CB	2C	CB	2C	83	2C	16
2D	B コ	2D	62	2D	42	2D	42	2D	62	2D	BA	2D	BA	2D	84	2D	02
2E	N ミ	2E	6E	2E	4E	2E	4E	2E	6E	2E	D0	2E	D0	2E	85	2E	05
2F	M モ	2F	6D	2F	4D	2F	4D	2F	6D	2F	D3	2F	D3	2F	86	2F	0D

キーコード		通常		SHIFT		CAPS		CAPS + SHIFT		カナ		カナ + SHIFT		GRPH		CTRL	
30	, < ネ 、	30	2C	30	3C	30	2C	30	3C	30	C8	30	A4	30	87		
31	. > ル 。	31	2E	31	3E	31	2E	31	3E	31	D9	31	A1	31	88		
32	/ ? メ ・	32	2E	32	3F	32	2F	32	3F	32	D2	32	A5	32	97		
33	_ ロ			33	5F			33	5F	33	D8	33	D8			33	1F
34	SPACE	34	20	34	20	34	20	34	20	34	20	34	20	34	20	34	20
35	XFER	35	00	35	00	35	00	35	00	35	00	35	00	35	00	35	00
36	ROLL UP	36	00	36	00	36	00	36	00	36	00	36	00	36	00	36	00
37	ROLL DOWN	37	00	37	00	37	00	37	00	37	00	37	00	37	00	37	00
38	INS	38	00	38	00	38	00	38	00	38	00	38	00	38	00	38	00
39	DEL	39	00	39	00	39	00	39	00	39	00	39	00	39	00	39	00
3A	↑	3A	00	3A	00	3A	00	3A	00	3A	00	3A	00	3A	00	3A	00
3B	←	3B	00	3B	00	3B	00	3B	00	3B	00	3B	00	3B	00	3B	00
3C	→	3C	00	3C	00	3C	00	3C	00	3C	00	3C	00	3C	00	3C	00
3D	↓	3D	00	3D	00	3D	00	3D	00	3D	00	3D	00	3D	00	3D	00
3E	HOME CLR	3E	00	3E	00	3E	00	3E	00	3E	00	3E	00				
3F	HELP	3F	00	3F	00	3F	00	3F	00	3F	00	3F	00	3F	00	3F	00

キーコード		通常		SHIFT		CAPS		CAPS + SHIFT		カナ		カナ + SHIFT		GRPH		CTRL	
40	−	40	2D	40	2D	40	2D	40	2D	40	2D	40	2D	40	2D	40	2D
41	/	41	2F	41	2F	41	2F	41	2F	41	2F	41	2F	41	2F	41	2F
42	7	42	37	42	37	42	37	42	37	42	37	42	37	42	37	42	37
43	8	43	38	43	38	43	38	43	38	43	38	43	38	43	38	43	38
44	9	44	39	44	39	44	39	44	39	44	39	44	39	44	39	44	39
45	*	45	2A	45	2A	45	2A	45	2A	45	2A	45	2A	45	2A	45	2A
46	4	46	34	46	34	46	34	46	34	46	34	46	34	46	34	46	34
47	5	47	35	47	35	47	35	47	35	47	35	47	35	47	35	47	35
48	6	48	36	48	36	48	36	48	36	48	36	48	36	48	36	48	36
49	+	49	2B	49	2B	49	2B	49	2B	49	2B	49	2B	49	2B	49	2B
4A	1	4A	31	4A	31	4A	31	4A	31	4A	31	4A	31	4A	31	4A	31
4B	2	4B	32	4B	32	4B	32	4B	32	4B	32	4B	32	4B	32	4B	32
4C	3	4C	33	4C	33	4C	33	4C	33	4C	33	4C	33	4C	33	4C	33
4D	=	4D	3D	4D	3D	4D	3D	4D	3D	4D	3D	4D	3D	4D	3D	4D	3D
4E	0	4E	30	4E	30	4E	30	4E	30	4E	30	4E	30	4E	30	4E	30
4F	,	4F	2C	4F	2C	4F	2C	4F	2C	4F	2C	4F	2C	4F	2C	4F	2C

キーコード		通常		SHIFT		CAPS		CAPS + SHIFT		カナ		カナ + SHIFT		GRPH		CTRL	
50	・	50	2E	50	2E	50	2E	50	2E	50	2E	50	2E	50	9E	50	2E
51	NFER	51	00	A1	00	51	00	A1	00	51	00	A1	00	51	00	B1	00

キーコード		通常	SHIFT	CAPS	CAPS + SHIFT	カナ	カナ + SHIFT	GRPH	CTRL
60	STOP								
61	COPY								
62	f・1	62 00	82 00	62 00	82 00	62 00	82 00		92 00
63	f・2	63 00	83 00	63 00	83 00	63 00	83 00		93 00
64	f・3	64 00	84 00	64 00	84 00	64 00	84 00		94 00
65	f・4	65 00	85 00	65 00	85 00	65 00	85 00		95 00
66	f・5	66 00	86 00	66 00	86 00	66 00	86 00		96 00
67	f・6	67 00	87 00	67 00	87 00	67 00	87 00		97 00
68	f・7	68 00	88 00	68 00	88 00	68 00	88 00		98 00
69	f・8	69 00	89 00	69 00	89 00	69 00	89 00		99 00
6A	f・9	6A 00	8A 00	6A 00	8A 00	6A 00	8A 00		9A 00
6B	f・10	6B 00	8B 00	6B 00	8B 00	6B 00	8B 00		9B 00

キーコード		通常	SHIFT	CAPS	CAPS + SHIFT	カナ	カナ + SHIFT	GRPH	CTRL
70	SHIFT								
71	CAPS								
72	カナ								
73	GRPH								
74	CTRL								

≡6.2≡ キーボードのI/O制御命令

キーボードの制御用に割り当てられているI／Oポートアドレスは2種類あって，41Hと43Hです．このI／Oポートを介して制御データやパラメータを入出力することにより，キーボードの制御を行っています．キーボードに関するI／O制御命令を表1-7に示します．個々の命令の説明を以下に述べます．

なお，キーボードのI／Oはシステムによって初期設定／利用されているので，ユーザがむやみに操作すると動作がおかしくなることがあります．システムは，キーが押されると割り込みによって押されたキーをバッファにため込むので，キーボードのI／Oを用いる場合には割り込みなどに関する十分な知識が必要です．機械語でプログラムを作る場合でも，キーボードBIOSの機能で不足することはまずないはずですから，BIOSでキー入力する方がよいでしょう．

表1-7 キーボードのI/O制御命令

I/O制御命令	I/Oポートアドレス	I/O	制御データ								解説
			b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0	
モードライト	43H	OUT	S2	S1	P	PEN	L2	L1	B2	B1	シリアルインタフェースμPD8251Aの動作モードを初期設定する．
コマンドライト	43H	OUT	×	IR	KB	ER	RST	RX	RTY	TX	モードライト命令実行後に行う．μPD8251Aの動作を指定する．
データリード	41H	IN	←── 受信データ ──→								キーボードからμPD8251Aに送られて来たデータを1バイト分読み込む．
ステータスリード	43H	IN	×	×	FE	OE	PE	×	RDY	×	μPD8251Aのステータス情報を読みこむ．

(1)モードライト命令

［機能］

シリアルインターフェースμPD8251Aの初期化を行います．ただし，μPD8251Aの内部または外部リセット動作後に続く必要があります．

ボーレート	B_2B_1＝	00：同期モード 10：×16モード	01：×1モード 11：×64モード
キャラクタ長	L_2L_1＝	00：5ビット 10：7ビット	01：6ビット 11：8ビット
パリティ	PEN＝ P　＝	0：ディスエーブル 0：ODD	1：イネーブル 1：EVEN
ストップビット	S_2S_1＝	00：無効 10：1.5ビット	01：1ビット 11：2ビット

(2)コマンドライト命令

［機能］

シリアルインターフェースμPD8251Aの動作を指定します．この命令はモードライト命令実行後に，受け付け可能になります．ただし，一度モードライトが行われると，それ以降はすべてコマンドライトとして受けとられます．

送信	TX ＝	0：ディスエーブル	1：イネーブル
リトライ	RTY＝	0：イネーブル	1：ディスエーブル
受信	RX ＝	0：ディスエーブル	1：イネーブル
リセット	RST＝	0：ディスエーブル	1：イネーブル
エラーリセット	ER ＝	0：　――	1：エラーフラグPE, OE, FEをリセットする
KB送信	KB ＝	0：イネーブル	1：ディスエーブル
内部リセット	IR ＝	0：　――	1：モードライト命令受付状態に戻す

(3)ステータスリード命令

［機能］

μPD8251Aのステータス情報を読み込む．

	RDY＝	インタフェース信号$\overline{RDY}$と同じ値	
パリティエラー	PE ＝	0：――	1：パリティエラー検出
オーバーランエラー	OE ＝		1：
フレーミングエラー	FE ＝	0：――	1：ストップビット未検出

≡6.3≡ キーボードBIOS

キーボード（KB）用BIOSについて説明します．キーボードBIOSは，キーボードのI／O制御をより簡単な手続きで実現するために用意されている基本ソフトウェアです．BIOS一般についての概要については，第1章2.2あるいは第3章6を参照してください．

キーボードBIOSは，**表1-8**に示すように5種類のキーボードBIOSコマンドとして系統化されていますが，これらのコマンドを実行するための手続きは，以下に示す通りです．

①レジスタAHにキーボードBIOSコマンドコードを設定する
②必要ならば所定のレジスタおよびメモリ領域にパラメータ値を設定しておく
③ソフトウェア割り込みの実行

INT 18H（キーボードBIOSルーチンのベクタコードは18H）

表1-8 キーボード BIOSコマンド

BIOSコマンド名称	BIOSコマンドコード	説明
KINT	03H	キーボードインターフェースの初期化
READ	00H	キーコードデータの読み出し
BSENS	01H	キーコードバッファ状態の検査
KSENS1	02H	シフトキー押下状態の検査
KSENS2	04H	キー押下状態の検査

(1)キーボードインターフェースの初期化コマンド(KINT)

［機能］

ⓐ キーボードインターフェースとして用いている汎用シリアルインターフェースμPD8251Aを初期化する。

ⓒ システム共通エリアにおいて，キーボードBIOSが使用するブロックを初期化する。

［割り込みコード］

INT 18H

［コマンドコード］

AH←03H

キーボードBIOSが使用するシステム共通エリア内のブロック（表2-2参照）

相対アドレス	サイズ(バイト)	ブロック名	内容
102H	32	KB_BUF	キーコードデータバッファ
122H	2	KB_TBL	キーコード変換テーブルのオフセットアドレス
124H	2	KB_HEAD	キーコードデータバッファの格納済エリアの先頭オフセットアドレス
126H	2	KB_TAIL	キーコードデータバッファ格納済エリアの最終アドレス+1
128H	1	KB_COUNT	キーコードデータバッファの格納済キーコード数
129H	1	KB_RTRY	エラーリトライカウンタ
12AH	16	KB_STS	キー入力状態テーブル
13AH	1	KB_SHFT	シフトキー状態テーブル

ベースアドレスは DS＝0040H

⑵キーコードデータの読み出しコマンド(READ)

［機能］

キーコードデータバッファ*の先頭にあるキーコードデータを読み出し，バッファを更新する．バッファが空のときは，データ待ち状態となる．

［割り込みコード］

INT 18H

［コマンドコード］

AH←00H

［出力］

AH←スキャンコード ｝両者をまとめてキーコードデータと呼ぶ
AL←内部コード　　 ｝キーコードデータはP36からの表1-6(2)を参照

リスト1-1
キーボードから入力されたキー内容を
CRTに表示するプログラム

```
                                ;
                                ; KB->CRT
                                ;
                                CSEG
                                ORG 0H
                                ;
0000 B86000                     MOV AX,60H
0003 8ED8                       MOV DS,AX
0005 8ED0                       MOV SS,AX
                                LOOP:
0007 33C0                       XOR AX,AX          ;READ COMMAND
0009 CD18                       INT 018H           ;KBIOS CALL
                                ;
000B 3C08                       CMP AL,8
000D 7408           0017        JE OUT
                                ;
000F BF3D00                     MOV DI,3DH
0012 CDC4                       INT 0C4H           ;N88-SYSTEM CALL
0014 E9F0FF         0007        JMP LOOP
                                OUT:
0017 CF                         IRET
                                ;
                                END
```

＊ キーコードバッファはシステム共通エリアに存在する．表2-3(2)参照．

⑶キーコードデータバッファ状態の検査(BSENS)

[機能]

READとほぼ同じだが，バッファは更新されない．

[割り込みコード]

INT 18H

[コマンドコード]

AH←01H

[出力]

AH←スキャンコード｝両者をまとめてキーコードデータを呼ぶ

AL←内部コード　　｝キーコードデータはP36からの表1-6(2)を参照

BH←00H（無効データのとき）／01H（有効データのとき）

⑷シフトキー押下状態の検査(KSENS 1)

[機能]

シフトキー（SHIFT，CAPS，カナ，GRPH，CTRL）の押下状態を調べる

[割り込みコード]

INT 18H

[コマンドコード]

AH←02H

[出力]

AL← | 0 0 0 b_4 b_3 b_2 b_1 b_0 | （押下状態のときビット値＝1）

シフトキーとビット番号の対応

b_0	SHIFT
b_1	CAPS
b_2	カナ
b_3	GRPH
b_4	CTRL

⑸キー押下状態の検査(KSENS 2)

[機能]

9801のキーボード上のすべてのキーのうち，現在どのキーが押されているか，離されているかを調べる．

[割り込みコード]

INT 18H

［コマンドコード］

AH←04H

［入力］

AL←キーコードグループ番号＊(00H～0FH)

［出力］

AH← $b_7 b_6 b_5 b_4 b_3 b_2 b_1 b_0$ (押下状態の時ビット値＝1)

指定されたキーコード・グループに属する8個のキーの押下状態が返される.

表1-9 キーコードグループと各キーの対応

キーコードグループ番号↓

	bit0	bit 1	bit2	bit3	bit4	bit5	bit6	bit7
00H	ESC	1!ヌ	2"フ	3#ア	4$ウ	5%エ	6&オ	7'ヤ
01H	8(ユ	9)ヨ	0ワ	-=ホ	^へ	¥|-	BS	TAB
02H	Qタ	Wテ	Eイ	Rス	Tカ	Yン	Uナ	Iニ
03H	Oラ	Pセ	@゛	[{゜	復改	Aチ	Sト	Dシ
04H	Fハ	Gキ	Hク	Jマ	Kノ	Lリ	;レ	:ケ
05H	]}ム	Zツ	Xサ	Cソ	Vヒ	Bコ	Nミ	Mモ
06H	,<ネ	.>ル	/?メ	_ロ	空白	XFER	RLUP	RLDN
07H	INS	DEL	↑	←	→	↓	HOME	HELP
08H	−	／	7	8	9	＊	4	5
09H	6	＋	1	2	3	＝	0	,
0AH	.	NFER						
0BH								
0CH	STOP	COPY	f･1	f･2	f･3	f･4	f･5	f･6
0DH	f･7	f･8	f･9	f･10				
0EH	SHFT	CAPS	カナ	GRPH	CTRL			
0FH								

例えば, CAPSとカナが押されているとき「AL＝0EH」としてこのコマンドを実行するとAHには06H(LSB 011000000 MSB)の値が返ってくる.

＊ 00H～6BHまでのキーコードを, 小さい方から8個ずつに分類して, それぞれに番号0, 1, 2, …, 0FHを与えたもの.

7 タイマ

7.1 概要

PC-98システムには，タイマと呼ばれるLSI(プログラマブル・カウンタ／タイマμPD8253)が内蔵されています．

タイマは，3組の16ビットカウンタ(#0〜2)で構成されていて，各カウンタとも6種類の動作モード(モード0〜5)を設定できます．

ただし，PC-98では8253のGATE端子がプルアップされているため，動作モード1，モード5はまったく使用できません．

各々**表1-10**の目的に使用されています．

表1-10 各カウンタの用途

カウンタ名	用　　途
カウンタ#0	インターバルタイマ（モード3で使用）
カウンタ#1	スピーカ周波数*（モード3で使用） (U/VF/VM/UV/VX) メモリリフレッシュ (E/F/M)
カウンタ#2	RS-232C

* スピーカON／OFF制御命令
（BIOSコールとI／Oアクセスの2種を示すが，なるべくBIOSコールを用いた方がよい）

方　法	ON	OFF
BIOSコール	MOV　AH, 17H INT　18H	MOV　AH, 18H INT　18H
I/O直接アクセス	MOV　AL, 06H OUT　37H, AL	MOV　AL, 07H OUT　37H, AL

≡7.2≡ タイマのI/O制御命令

タイマに関するI／O制御命令について説明します．

タイマの制御用に割り当てられているI／Oポートアドレスは4種類あり，71H，75H，77H，3FDBH, 3FDFHです．このI／Oポートを介して，制御データやパラメータを入出力することにより，タイマの制御を行っています．

タイマに関するI／O制御命令を表1-11にまとめて示します．個々の命令の説明を以下に述べます．

表1-11 タイマのI/O制御命令

I/O制御命令	I/Oポートアドレス	制御データ（バイト）	説明
WRITE＃0	71H	2	カウンタ＃0に値を設定する
READ＃0	71H	2	カウンタ＃0の値を読み出す
WRITE＃1	3FDBH*	2	カウンタ＃1に値を設定する
READ＃1	3FDBH*	2	カウンタ＃1の値を読み出す
WRITE＃2	75H	2	カウンタ＃2に値を設定する
READ＃2	75H	2	カウンタ＃2の値を読み出す
MODE	77H**	1	各カウンタ＃nの動作モードを設定する

＊　E/F/Mでは73H（他機種とは異なる）
＊＊　U/VF/VM/UV/VX/UXでは3FDFHも可

なお，カウンタ値や動作モードはシステムにより自動的に初期化されるのでユーザが特に設定を行う必要はありません．また，カウンタ＃0，＃2はBIOSによって設定（カウンタ＃0は1章7.3タイマBIOS，カウンタ＃2は6章2.4参照）されるので，ユーザは直接I／Oを操作しない方がよいでしょう．

カウンタ＃1（U／VF／VM／UV／VX／UX）はリセット時には2kHzを出力していますが，カウンタ値を変更すれば，ブザーの音程を変えることができます．E／F／Mでは，ブザーの音程を変えることはできません．

(1)MODE

[機能]

各カウンタ#nの動作モードを設定するための1バイトデータを送出する.

MSB　　　　　　　　　　　　LSB

SC1	SC0	RW1	RW0	M2	M1	M0	BCD

	ビット名	解説	
カウンタの選択	SC_1SC_0 =	00：カウンタ#0 10：カウンタ#2	01：カウンタ#1 11：使用不可
データの選択	RW_1RW_0 =	00：カウント・ラッチ 10：上位バイト指定	01：下位バイト指定 11：下位,上位の順
モード選択*	$M_2M_1M_0$ =	000：モード0 ×11：モード3	×10：モード2
数表現	BCD =	0：バイナリ表現	1：BCD表現

*

①モード0

[機能]

カウント数を設定すると,デクリメントを開始し,終了した時点で出力信号の値が0から1へ変わる.次のカウント数を設定するまで出力信号は変化しない.インターバルタイマとして動作する.

②モード2

[機能]

1カウンタ設定値nに応じて入力クロックをn分周し,デューティ比1/nの波形を出力する.レートジェネレータとして動作する.

③モード3

[機能]

方形波レートジェネレータとして動作する.モード2と同様に入力クロックをn分周するが,カウント数が偶数の場合のデューティ比は$\frac{1}{2}$,奇数の場合のデューティ比は(n－1)/2nである.

(2)WRITE#0

[機能]

カウンタ#0に16ビット値nを設定する.

カウンタ#0をモード0でインターバルタイマとして使用する場合,設定値nとウエイト時間の関係は下記の通りです.

クロック(MHz)	ウエイト時間(msec)	備考
10	n×1/2457.6	
8	n×1/1996.8	

(3)WRITE＃1 (U/VF/VM/UV/VX/UXのみ)*

［機能］

カウンタ#1に16ビット値nを設定する．

カウンタ#1をモード3で方形波レートジェネレータとして動作させ，スピーカ周波数の基準とする場合，設定値nと周波数の関係は下記の通りです．

クロック(MHz)	スピーカ周波数(KHz)	備　　考
10	2457.6×1/n	n＝1229(既定値)
8	1996.8×1/n	n＝998(既定値)

(4)WRITE＃2

［機能］

カウンタ#2に16ビット値nを設定する．

カウンタ#2をモード2でレートジェネレータとして動作させ，RS-232Cインターフェースの通信速度を基準とする場合，設定値nと通信速度の関係は下記の通りです．ただし，通信プロトコルは調歩同期式1／16モードであるとします．

通信速度(ボー)	設定値 n	
	5MHz/10MHz時	8MHz時
9600	16	13
4800	32	26
2400	64	52
1200	128	104
600	256	208
300	512	416
150	1024	832
75	2048	1664

(5)READ＃0(＃1，＃2)

［機能］

カウンタ#0（#1，#2）のカウント値を読み出す．

---

* E/F/Mではメモリリフレッシュ用にシステムで使用しているので，下記値に固定しておき，変更しない｛ 5MHz　n＝70
　　　8MHz　n＝57

7.3 タイマBIOS

タイマBIOSについて説明します．タイマBIOSは，インターバルタイマ（カウンタ#0）を，より簡単な手続きで活用できるように用意されている基本ソフトウェアです．BIOS一般についての概要については第1章2.2，第3章5を参照してください。

タイマBIOSコマンドを実行するための手続きは，以下に示す通りです．

①レジスタAHにタイマBIOSコマンドコードを設定する

AH←02H（タイマBIOSコマンドは1種類のみ）*

②所定のレジスタにパラメータ値を設定する

③ソフトウェア割り込みの実行

INT 1CH（タイマBIOSルーチンのベクタコードは1CH）

タイマBIOSコマンドについての解説を以下に述べます．

インターバルタイマの設定

［機能］

インターバルタイマ値を設定し，起動させる．設定値に相当する時間経過後，指定した戻り番地にリターンする．

［割り込みコード］

INT 1CH

［コマンドコード］

AH←02H

［入力］

CX←インターバルタイマ値n（0≦n≦65535）

注）設定時間＝10msec×n（ただしn＝0のときは655360msec）

ES←戻り番地（セグメントアドレス）

BX←戻り番地（オフセットアドレス）

---

* 本来，タイマBIOSは，第1章8.4で述べるカレンダBIOSにまとめて扱われており，同じベクタコード1CHが割り当てられている．本書では，構成上の理由で別々に扱っている．

[インターバルタイマの使用例]

9000：0000番地にあるルーチンを１分後に呼び出す場合を示します．

リスト1-2

```
;●タイマの設定ルーチン
;（このルーチン実行１分後に割込が起こり，9000:0000番地に実行が移る）
        PUSH    AX
        PUSH    BX
        PUSH    CX
        PUSH    ES
        MOV     AX,9000H        ;割り込み先アドレス（segment 9000h）
        MOV     ES,AX
        MOV     BX,0000H        ;割り込み先アドレス（offset  0000h）
        MOV     CX,60*100       ;60*100*10msec後
        MOV     AH,02H          ;インターバルタイマＢＩＯＳコール
        INT     1CH
        POP     ES
        POP     CX
        POP     BX
        POP     AX
        RET

;●割り込み先ルーチン
9000:0000       PUSH    AX      ;処理ルーチンで使用されるレジスタは
                PUSH    BX      ;全て退避する
                PUSH    CX
                PUSH    DX
                PUSH    SI
                PUSH    DI
                PUSH    DS
                PUSH    ES
                    |
                    |           ;実行したい処理を行う
                    |
                POP     ES      ;レジスタを復帰させる
                POP     DS
                POP     DI
                POP     SI
                POP     DX
                POP     CX
                POP     BX
                POP     AX
                IRET            ;IRET命令で割込処理ルーチンを終える
```

8 カレンダ時計

8.1 概要

PC-9801はカレンダと時計の機能を内蔵しています．BASICではDATE＄，TIME＄という関数が用意されていて，日付と時刻の設定・読み出しが容易にできます．「1988年2月3日12時10分」を設定するには，BASICのダイレクトモードで，

DATE＄＝"88／02／03"

TIME＄＝"12：10：00"

と入力すれば，リターンキーを押した時点で設定されて，以降，時刻を刻み続けます．カレンダ時計の機能はバッテリーでバックアップされているので，本体の電源をOFFにしても約2ヵ月は動作し続けます．

現在の日付，時刻を表示させたければ，PRINT DATE＄，TIME＄［CR］でよいわけです．

PC-9801E／F／M／U／VM／VF／UVのカレンダ時計には，μPD1990Aという時計用LSIが使われています．μPD1990Aは月，日，曜，時，分，秒のデータを管理し，年はシステムの不揮発性メモリ（A3FFEH番地）に書き込まれています．このため，月の大小は自動的に判断されますが，年の繰上がりや閏年の判別は自動的には行われません（毎年正月には9801に年越しの設定を，閏年の2月29日には29日の設定をしなければなりません．ちなみに，μPD1990Aの日付は2月29日に設定することができます）．なお，曜日のデータはBASICでは使用していません．

PC-9801VM21／UV21／VX／UXにはμPD4990AというμPD1990Aの改良版の時計用LSIが使われています． μPD4990AにはμPD1990Aの欠点であった，年の繰上がりと閏年の自動判別機能が備わりました．このため，VM21/UV21／VX／UXでは時計が狂ったり，長時間電源を入れないでバッテリーが切れてしまったりしない限り，日付の設定は一度だけすればよいことになります．

≡8.2≡ カレンダ時計のI/O制御命令

カレンダ時計のI／O制御命令について説明します．μPD1990A／4990Aに割り当てられたI／Oポートは2種類あって，そのアドレスは20H，33Hです．このI／Oポートアドレスを介して制御データを入出力することにより，μPD1990A／4990Aの制御を行っています．第1章4.1のI／Oポートアドレス一覧表を見ると，μPD1990A／4990Aのポートアドレスは20Hだけですが，システムポートの中の8255AポートBビット0は時計LSIのデータ出力につながれています．

μPD1990A／4990AはつながれているI／Oポートのアドレス等は同じですが，4990Aを搭載している機種では4990Aを拡張モードで使用しているので，1990A互換の制御命令を与えると時刻などの情報が揮発してしまいます．よって，時刻の読出／設定をする場合は必ずBIOSを使うようにします．

カレンダ時計のI／O制御命令を**表1-12**に示します．

表1-12 カレンダ時計のI/O制御命令

I/O 制御命令	I/Oポート アドレス	I/O	制御データ							機能説明
			b_7	b_6	b_5	b_4	b_3	b_2	b_1	
セット レジスタ	20H	OUT	×	×	DI	CLK	STB	C_2	C_1	C_0	コマンドのセット及び時刻設定のためのデータの書き込みを行う
リード データ	33H	IN	×	×	×	×	×	×	×	DO	μPD1990から時刻を読み出す

○×印のビットは不定

≡8.3≡ μPD1990Aの制御方法

参考までにμPD1990Aの制御方法を示します．

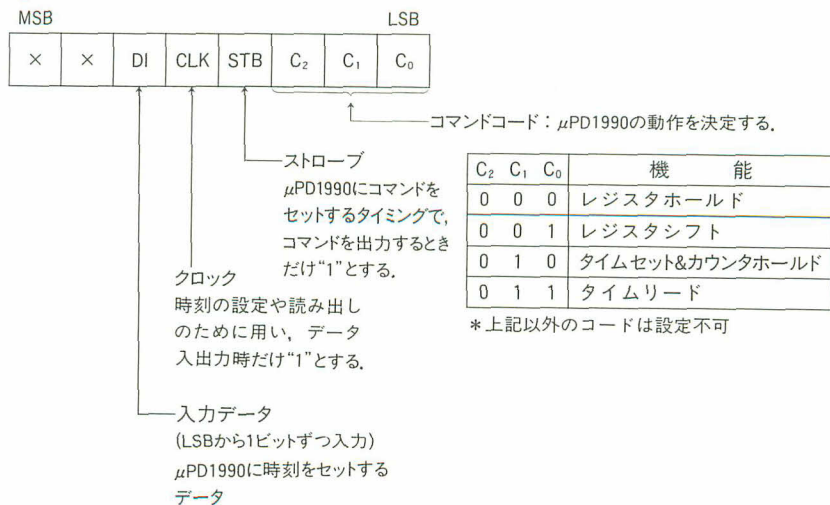
(1)セットレジスタ

セットレジスタ命令は，μPD1990に各種のコマンドや時刻設定のデータを出力します（**図1-11**参照）．

μPD4990Aは，1990Aのテストモードを廃して拡張モードにしている．

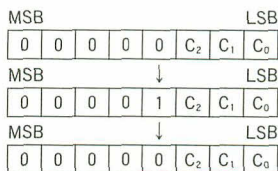
図1-11 セットレジスタの制御データ

C_2	C_1	C_0	機　　能
0	0	0	レジスタホールド
0	0	1	レジスタシフト
0	1	0	タイムセット&カウンタホールド
0	1	1	タイムリード

＊上記以外のコードは設定不可

※コマンドのセット

①コマンドコードをセットする.
↓(2μs以上の間隔)
②STBを1にする.
↓(4μs以上の間隔)
③STBを0に戻す.
↓(2μs以上の間隔)
④(次の処理へ)

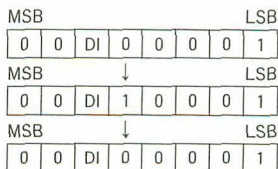
MSB							LSB
0	0	0	0	0	C_2	C_1	C_0
0	0	0	0	1	C_2	C_1	C_0
0	0	0	0	0	C_2	C_1	C_0

この間 C_2, C_1, C_0 の設定値を変化させない.

※時刻の設定

①シフトレジスタコマンドをセットする.
(1ビットずつ設定するため)
↓
②入力データ DI をセットする.
↓(2μs以上の間隔)
③CLK を1にする.
↓(2μs以上の間隔)
④CLK を0に戻す.

MSB							LSB
0	0	DI	0	0	0	0	1
0	0	DI	1	0	0	0	1
0	0	DI	0	0	0	0	1

この間 DI の設定値を変化させない.

↓
⑤ ②～④を40ビットのデータについて繰り返す.
↓
⑥タイムセット & カウンタホールドコマンドをセットする.
↓
(時刻設定のためのタイミング待ちループ)
↓
⑦レジスタホールドコマンドを設定する.
この時点で時刻設定がなされ，刻時動作を始める.

(2)リードデータ

リードデータ命令は，現在の時刻のデータを図1-12の形式で，LSBから1ビットずつ読み出します．

図1-12 リードデータの制御データ

カレンダ時計の40ビットのデータがLSBから1ビットずつ出力される．

※時刻の読み出し

①タイムリードコマンドをセットする．

↓

②シフトレジスタコマンドをセットする．

↓（40μs以上の間隔）

MSB							LSB
0	0	0	1	0	0	0	1

↓

MSB							LSB
0	0	0	0	0	0	0	1

⑥ ③～⑤を40ビットのデータについて繰り返す．

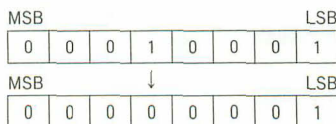



図1-13 カレンダ時計入出力データ形式

8.4 カレンダ時計のBIOS

カレンダ時計のBIOSについて説明します．BIOSを利用すれば，機種による時計LSIの違い（μPD1990AとμPD4990A）を区別する必要がありませんし，また容易に日付と時刻を設定・読み出しできます．

カレンダ時計のBIOSを実行するための手続きは，

①AHレジスタにBIOSコマンドコードを設定

↓

②必要に応じてほかのレジスタや所定のパラメータリスト領域に値を設定

↓

③ソフトウェア割り込み1CH（INT　1CH）を実行

となります．

(1)日付・時刻の読み出し

［機能］

現在の日付（年，月，曜，日），時刻（時，分，秒）を読み出す．

［割り込みコード］

INT 1CH

［コマンドコード］

AH←00H

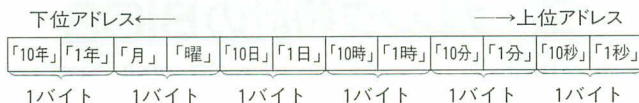
［入力］

ES←日付・時刻のデータを受け取るバッファ（6バイト）のセグメント・アドレス

BX←ESに対応するオフセット・アドレス

［出力］

ES：BXで指定されたアドレスのメモリ上に図1-14に示す形式で，現在の日付・時刻が読み出されます．

図1-14　データバッファ形式

実際にプログラムを実行してみましょう（**リスト1－3**）．

リスト1-3
日付・時刻の読み出し

```
CLEAR ,&H1F00
Ok
DEF SEG=&H1F00
Ok
h]L0000,000E
0000 B8001F        MOV    AX,1F00  ] ESに1F00Hを設定
0003 8EC0          MOV    ES,AX    ]
0005 B80001        MOV    AX,0100  ] BXに100Hを設定
0008 89C3          MOV    BX,AX    ]
000A B400          MOV    AH,00   ―― AHに00Hを設定
000C CD1C          INT    1C      ―― BIOSのコール
000E F4            HLT
h]G0000,000E
*1F00:000E
h]D100,105
0100 86 25 21 13 22 11
     '86年 | 21日 13時 22分 11秒
       2月・金曜
```

EX，BXで指定したメモリ上に6バイトのデータが読み出されました．

⑵日付・時刻の設定

［機能］

年，月，曜日，日，時，分，秒を設定して，計時を開始する．

［割り込みコード］

INT ←1CH

［コマンドコード］

AH←01H

［入力］

ES←設定する日付・時刻のデータを格納するバッファ（6バイト）のセグメント・アドレス

BX←ESに対応するオフセット・アドレス．

＊ ES：BXで指定したアドレスのメモリ上に日付・時刻のデータを**リスト1－3**と同じ形式で格納します．

1986年2月14日金曜日の12時10分00秒を設定してみます(**リスト1－4**)．

リスト1-4
日時・時刻の設定

```
CLEAR ,&H1F00
Ok
DEF SEG=&H1F00
Ok
0000 B8001F        MOV    AX,1F00    ] ESに1F00Hを設定
0003 8EC0          MOV    ES,AX      ]
0005 B80001        MOV    AX,0100    ] BXに100Hを設定
0008 89C3          MOV    BX,AX      ]
000A B400          MOV    AH,00   ―― AHに01Hを設定
000C CD1C          INT    1C      ―― BIOSのコール
000E F4            HLT
h]S100
0100 00-86 00-25 00-14 00-12 00-10 00-00
        '86年 2月・金曜 14日  12時  10分  00秒
```

このようにデータを設定したうえでプログラムを実行すればよいわけです．

```
h] G0000, 000E
＊1F00,
```

9 DMAコントローラ

9.1 DMAコントローラの概要

通常，周辺LSI（周辺装置）とメモリの間のデータ転送はCPUが制御しますが，そのデータ量が大きいときにはCPUに負荷がかかりすぎます．そこで，データ転送専用のLSIとしてDMAコントローラ（μPD8237A）が用意されています．DMAコントローラを用いれば，CPUを介さないでデータ転送を高速で行うことができます．

DMAコントローラは，4つの独立したチャネルを持っていて，各チャネルともデータの転送幅は8ビットパラレルです．各チャネルは，**表1-12**に示すように各種の周辺装置に割り当てられています．なお，各チャネルには優先順位が定められています．

表1-12からも明らかなように，PC-98システムでは，DMAコントローラを専らディスク装置に対する入出力制御用に使用しています．そのため，DMAコントローラ（μPD8237A）自体は何種類もの転送モードで使用できるのですが，そのなかで，"シングル転送モード"に限定して使用しています．このモードでは，1回のDMA要求に対して，転送動作を1回だけ実行して終了します．1回の動作モードで転送するデータは，1バイトです．

表1-12　DMAコントローラのI／Oチャネル

チャネル番号	優先順位	用途	拡張スロット番号				
			E	F1,2/VF/VM	M2	M3	F3/U/UV
0	1	5"ハードディスク	＃1～＃6	＃1～＃4	＃1～＃4	＃1～＃2	＃1～＃2
1	2	メモリリフレッシュ*	――	――	――	――	――
2	3	1MB FDD	＃6	＃4	専用コネクタ	専用コネクタ	＃2
3	4	640KB FDD	＃1～＃5	＃1～＃3	＃1～＃3	＃1，＃2	＃1

注）メモリリフレッシュは64KB単位で行う
　* VX/UXでは使用せず

9.2 DMAコントローラのI/O制御命令

DMAコントローラのＩ／Ｏ制御命令について説明します．

DMAコントローラは，表１-13に示すような内部レジスタを持っていて，合計サイズは344ビットです．

DMAコントローラに割り当てられているＩ／Ｏポートアドレスは19種類で，表１-14に示すとおりです．これらのＩ／Ｏポートを介して制御データを入出力することにより，DMAコントローラの制御を行っています．

個々のＩ／Ｏ制御命令についての詳細を以下に述べます．

表1-13　DMAコントローラの内部レジスタ

レジスタ名称	サイズ(ビット)	本数	備　　考
コマンドレジスタ	8	1	DMAコントローラの動作を制御する８ビットの制御データを保持しておくためのレジスタ
モードレジスタ	6	4	４つのDMAチャネルのそれぞれの動作モードを指定する制御データを保持しておくためのレジスタ
リクエストレジスタ	4	1	DMAチャネルの動作開始を指定するデータを書き込むためのレジスタ（PC-98システムでは使用禁止）
マスクレジスタ	4	1	４つのDMAチャネルそれぞれについて，DMA要求受け付けの許可／禁止を制御するデータを保持しておくためのレジスタ
ステータスレジスタ	8	1	４つのDMAチャネルの動作状態を示すデータが保持されているレジスタ
カレントアドレスレジスタ	16	4	DMA転送の対象となるメモリのオフセットアドレス保持される．１回のDMA動作で１バイト転送するごとに値が＋１または－１される．
カレントカウントレジスタ	16	4	DMA転送するデータのサイズ（バイト）を指定されるカウンタで，１回のDMA動作で１バイト転送するごとに値が－１される．
テンポラリアドレスレジスタ	16	4	カレントアドレスレジスタと同じ値が保持されていて，DMA動作中でも支障なくI／O制御命令でアドレスを読み書きできる．
テンポラリカウントレジスタ	16	4	カレントカウントレジスタと同じ値が保持されていて，DMA動作中でも支障なくI／O制御命令でカウントを読み書きできる．
ベースアドレスレジスタ	16	4	カウントアドレスレジスタにI／O制御命令を書き込んだ初期値がそのまま保持されている．
ベースカウントレジスタ	16	4	カレントカウントレジスタにI／O制御命令を書き込んだ初期値がそのまま保持されている．
テンポラリレジスタ	8	1	メモリ間転送の際に，転送データ（８ビット）を一時的に保持しておくために使われるレジスタ

表1-14　DMAコントローラのI/O制御命令

I/O制御命令	I/Oポートアドレス	I/O	制御データ								備考
			b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0	
マスタクリア	1BH	OUT	×	×	×	×	×	×	×	×	ハードウェアリセットと同等
ライトコマンド	11H	OUT	KS	DS	WS	PR	TM	CE	AH	MM	コマンドレジスタに制御データを書き込む
ライトモード	17H	OUT	MS1	MS0	ID	AT	TP1	TP0	CS1	CS0	モードレジスタに制御データを書き込む
ライトリクエスト	13H	OUT	×	×	×	×	×	RB	CS1	CS0	使用禁止
ライトシングルマスク	15H	OUT	×	×	×	×	×	MK	CS1	CS0	マスクレジスタにマスクパターンを書き込む（ただし1チャネルごと）
ライトオールマスク	1FH	OUT	×	×	×	×	MB3	MB2	MB1	MB0	マスクレジスタにマスクパターンを書き込む（ただし4チャネルすべて）
クリアマスク	1DH	OUT	×	×	×	×	×	MK	CS1	CS0	マスクレジスタのマスクビットをすべてクリアする
リードステータス	11H	IN	RQ3	RQ2	RQ1	RQ0	TC3	TC2	TC1	TC0	ステータスレジスタの内容を読み出す
クリアバイトポインタ	19H	OUT	×	×	×	×	×	×	×	×	DMA転送したデータの値（8ビット）を保持しているテンポラリレジスタを読み込む
チャネル#0アドレス	01H	I/O	← アドレス → 下位(上位)アドレス								2回の実行で下位バイト，上位バイトの順に読み書きする
チャネル#0カウント	03H	I/O	← アドレス → 下位(上位)バイト								2回の実行で下位バイト，上位バイトの順に読み書きする
チャネル#1アドレス	05H	I/O	← カウント → 下位(上位)バイト								2回の実行で下位バイト，上位バイトの順に読み書きする
チャネル#1カウント	07H	I/O	← アドレス → 下位(上位)バイト								2回の実行で下位バイト，上位バイトの順に読み書きする
チャネル#2アドレス	09H	I/O	← アドレス → 下位(上位)バイト								2回の実行で下位バイト，上位バイトの順に読み書きする
チャネル#2カウント	0BH	I/O	← カウント → 下位(上位)バイト								2回の実行で下位バイト，上位バイトの順に読み書きする
チャネル#3アドレス	0DH	I/O	← アドレス → 下位(上位)バイト								2回の実行で下位バイト，上位バイトの順に読み書きする
チャネル#3カウント	0FH	I/O	← カウント → 下位(上位)バイト								2回の実行で下位バイト，上位バイトの順に読み書きする
チャネル#0バンク	27H	OUT	×	×	×	×	← バンク番号 →				バンクを指定する．図2-1参照
チャネル#2バンク	23H	OUT	×	×	×	×	← バンク番号 →				バンクを指定する．図2-1参照
チャネル#3バンク	25H	OUT	×	×	×	×	← バンク番号 →				バンクを指定する．図2-1参照
リードテンポラリ	1BH	IN	×	×	×	×	×	×	×	×	アドレスデータ，カウントデータの下位・上位バイトの選択を行う

(1)ライトコマンド命令

ライトコマンド命令は,DMAコントローラの動作を規定する制御データをコマンドレジスタに書き込むための命令です.制御データの内容は,下記のようになっています.

名称	ビット番号	ビット値=0	ビット値=1	備考 注
MM	b_0	メモリ間転送禁止	メモリ間転送許可	初期値=0
AH	b_1	チャネル#0アドレスホールド禁止	チャネル#0アドレスホールド許可	初期値=0
CE	b_2	コントローラ許可(DMA要求受け付け可)	コントローラ禁止(DMA要求受け付け不可)	(注1)
TM	b_3	通常タイミング	圧縮タイミング	初期値=0
PR	b_4	固定優先順位	回転優先順位	初期値=0
WS	b_5	遅れライト選択	拡張ライト選択	初期値=0
DS	b_6	DREQ* アクティブハイ	DREQアクティブロウ	初期値=1
KS	b_7	DACK** アクティブロウ	DACKアクティブハイ	初期値=0

注)PC-98システムの設定値

注1)チャネル#3に接続されている640KB FDコントローラはDMAイネーブル／ディスエーブル制御回路を持っていてPC-98では,FDがデータの入力動作を行うときのみDMAイネーブルとなるようにシテテム設計されている.

DMAコントローラは,DMAイネーブル,つまりコントロール許可状態(b_2=0)のときのみ,DMA要求を受け付けるので,640KB FDコントローラからの制御がDMAディスエーブルのときには,他のDMAチャネルを他の周辺装置で使用しても640KB FDは影響を受けない.

* DREQ=DMA request(DMAサービス要求信号)

** DACK=DMA acknowledge(DREQの受け付け完了信号)

(2)リードステータス命令

リードステータス命令は,ステータスレジスタのデータを読み込むためのもので,データの具体的な内容は下記のようになっています.

名称	ビット番号	ビット値=0	ビット値=1	備考
TC0	b_0	チャネル#0がTC*に未到達	チャネル#0がTCに到達	各チャネルがTCに到達するごとに,または,外部EOP**が入力されるごとに1にセットされる.
TC1	b_1	チャネル#1がTCに未到達	チャネル#1がTCに到達	
TC2	b_2	チャネル#2がTCに未到達	チャネル#2がTCに到達	
TC3	b_3	チャネル#3がTCに未到達	チャネル#3がTCに到達	
RQ0	b_4	チャネル#0が非リクエスト状態	チャネル#0がリクエスト状態	各チャネルがサービス要求を出したとき1にセットされる
RQ1	b_5	チャネル#1が非リクエスト状態	チャネル#1がリクエスト状態	
RQ2	b_6	チャネル#2が非リクエスト状態	チャネル#2がリクエスト状態	
RQ3	b_7	チャネル#3が非リクエスト状態	チャネル#3がリクエスト状態	

注)リセットもしくはリードステータス命令によって,各ビットは0にクリアされる.

* TC=Terminal count
カレントカウントレジスタの値がDMA動作ごとに減数されていき,0に達したときにTCに到達したという.全データの転送が完了したことを示す.

** EOP=End of Process.DMA動作の終了を示す信号

(3)ライトシングルマスク命令，ライトオールマスク命令 クリアマスク命令

ライトシングルマスク命令，ライトオールマスク命令で送出する制御データの説明を下記に示します．

上記命令はいずれも各チャネルをDMA要求受付状態にするかどうかを指定するための命令です．マスクビットがクリアされているチャネルだけがDMA要求を受け付けることができます．マスクがセットされているチャネルはDMA要求を無視します．ライトシングルマスク命令は，1つのチャネルだけを選んで，そのマスクビットを書き換える命令であり，他方の命令は4つのチャネルすべてのマスクビットを一度に書き換える命令です．各チャネルのマスクビットの内容は，4ビットのマスクレジスタに書き込まれます．

なお，クリアマスク命令は，4チャネル全部のマスクビットをクリアし，DMA要求受付可能にします．

ライトシングルマスク命令

名称	ビット番号	解説	備考
CS1,0	b_1b_0	=00 チャネル#0 =01 チャネル#1 =10 チャネル#2 =11 チャネル#3	マスクするチャネルの選択
MK	b_2	0：マスクビットクリア 1：マスクビットセット	

ライトオールマスク命令

名称	ビット番号	ビット値=0	ビット値=1	備考
MB0	b_0	マスクビットクリア	マスクビットセット	チャネル#0
MB1	b_1	マスクビットクリア	マスクビットセット	チャネル#1
MB2	b_2	マスクビットクリア	マスクビットセット	チャネル#2
MB3	b_3	マスクビットクリア	マスクビットセット	チャネル#3

注）オートイニシャライズ許可状態のときには，マスクビットはセットされない．(2)ライトモード命令参照．

(4)ライトモード命令

4つのDMAチャネルのそれぞれにモードレジスタが用意されていますが，ライトモード命令はそれらのレジスタに制御データを書き込むための命令です．制御データの内容は，下記のようになっています．

名称	ビット番号	解説	備考
CS1,0	b_0b_1	=00 チャネル#0　=01 チャネル#1 =10 チャネル#2　=11 チャネル#3	チャネル選択 (注1)
TP1,0	b_3b_2	=00 ベリファイ転送　=01 ライト転送(周辺→メモリ) =10 リード転送(メモリ→周辺)	
AT	b_4	0：オートイニシャライズ禁止　1：オートイニシャライズ許可	(注2)
ID	b_5	0：アドレスインクリメント選択　1：アドレスデクリメント選択	
MS1,0	b_7b_6	=01 シングル転送モード選択	他のモード不可(注3)

注1) 4つのDMAチャネルに独立に6ビットのモードレジスタがそれぞれ用意されている．ビットCS1，CS0で目的のチャネルのモードレジスタを選択している．

注2) オートイニシャライズ許可状態のときには，DMA動作終了ごとに自動的に初期設定がなされる．
①DMA動作の終了(EOP信号)後に，ベースアドレスレジスタとベースカウントレジスタの値がそれぞれカウントアドレスとカレントカウントレジスタに自動的に転記される．
②マスクビットはセットされず，再度DMA要求受け付け可能状態に戻る．

注3) 1回のDMA要求に対して1回(1バイト)の転送を行うモード．転送後にカウンタがデクリメント，アドレスがデクリメント(もしくはインクリメント)される．DMAコントローラには数種類の転送モードがあるが，PC-98システムではシングル転送モードに限定して使用している．

(5)チャネル＃nアドレス命令，チャネル＃nバンク命令

DMAチャネル#nがアクセスするメモリのアドレスを読み書きします．まず，チャネル#nバンク命令で，バンクの指定を行います*．バンクのメモリサイズは64Kバイトです．次に，チャネル#nアドレス命令でバンク内のオフセットアドレスを16ビットで読み書きします．

なお，チャネル#nアドレス命令を2回連続して実行することにより，下位バイト，上位バイトの順にデータが入出力されます．ただし，チャネル#nアドレスの実行に先立ち，クリアバイトポインタ命令を実行しておく必要があります**．

(6)チャネル＃nカウント命令

各チャネルには，カレントカウントレジスタとベースカウントレジスタがあり，いずれも16ビットです．カウントレジスタに対するデータの読み書きをチャネル#nカウンタ命令で行っていて，1回の命令で1バイトづつ下位バイトから順次転送します．ただし，チャネル#nカウンタ命令の実行に先立ち，クリアポインタ命令を実行しておく必要があります*．

1回のDMA要求につき，1バイトのデータ転送が行われますが，その際にカウント値がデクリメントされていきます．

---

* バンクの概念については，図2-1参照．

** クリアバイトポインタ命令を実行しないと，チャネル＃nアドレス命令の2回の実行で転送される2バイトのデータが，下位・上位の順に正しく対応しなくなる可能性がある．

第2章

メモリ

CONTENT

1 概要

PC-98のメモリアドレス空間は1Mバイトに及びますが，これがシステムの中で，どのように使われているかについて，その概略を見ていきます．

8086（V30を含む）CPUでは，1Mバイトものメモリ空間を扱えるにも関わらず，8ビットCPUのような感覚で利用できるように設計されています．これは，8ビットCPUに慣れた設計者が，容易に16ビットに移行できるようにという配慮からです．ここでは，このための「セグメント」という概念も説明します．

VX／UXで追加採用された80286CPUは，物理的には16Mバイトまでのメモリを取り扱うことができます．ただし，16Mバイトのメモリを利用する場合には，80286CPUを「プロテクト・モード」というモードにしなければなりません．ところが，プロテクト・モードは専用のOS（オペレーティング・システム）を用いなければ有効に使うことができないため，PC-98ではもっぱら8086互換モードである，「リアル・モード」で使っています．そこで，本書でも80286のプロテクト・モードについては割愛しています．リアルモードでは8086CPUと全く同様の動作をするので（リセット時を除く），8086と言った場合には80286のリアル・モードを含むものと理解してください．

2 CPUアドレス空間

本説では，バンク及びセグメントの概念について説明します．

2.1 バンク

PC-98に使われている8086CPUでは，20ビット幅のメモリバスを持っています．20ビットで表現できるアドレス空間は，2の20乗（=1048576≒10^6）ですから，1Mバイトになります．

これに対して，一般の8ビットCPUではメモリバスの幅は16ビットです．8ビットCPUと同じような感覚でプログラミングできるように設計された8086CPUでは，16ビット幅のアドレス指定方法で20ビットのメモリをアクセスする必要があります．そこで，「セグメント」という概念が考え出されました．

図2-1　CPUアドレス空間とバンクの概念

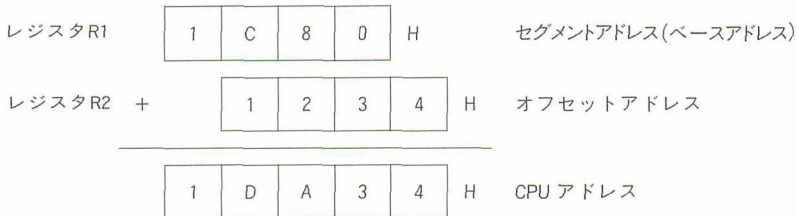
各バンクの先頭CPUアドレス	バンク番号	サイズ(B)	CPUアドレスの*相対アドレス表現 セグメントアドレス：オフセットアドレス
F0000H	FH	64K	F000H：0000H
E0000H	EH	64K	E000H：0000H
D0000H	DH	64K	D000H：0000H
C0000H	CH	64K	C000H：0000H
B0000H	BH	64K	B000H：0000H
A0000H	AH	64K	A000H：0000H
90000H	9H	64K	9000H：0000H
80000H	8H	64K	8000H：0000H
70000H	7H	64K	7000H：0000H
60000H	6H	64K	6000H：0000H
50000H	5H	64K	5000H：0000H
40000H	4H	64K	4000H：0000H
30000H	3H	64K	3000H：0000H
20000H	2H	64K	2000H：0000H
10000H	1H	64K	1000H：0000H
00000H	0H	64K	0000H：0000H

* それぞれ16ビットのセグメントアドレス，オフセットアドレスを組み合わせて，20ビットのCPUアドレスを表現している．詳細については本章2.3参照

≡2.2≡ セグメント

PC-98では，20ビットのCPUアドレスを，16ビットのレジスタを用いてどのように表現しているかを説明します．

PC-98では，2本の16ビットレジスタを組み合わせて，20ビットのCPUアドレスを指定しています．その様子を図2-2に示します．

図2-2 セグメントアドレス，オフセットアドレスとCPUアドレスの関係

一方の16ビットレジスタ(R1)でセグメントアドレスを指定し，他方の16ビットレジスタ(R2)の値でオフセットアドレスを指定します．この場合，レジスタ(R1)の値を4ビット分だけシフトさせた値がセグメントアドレスになります．つまり，レジスタ(R1)の値が1C80Hであれば，実際には1C800Hを意味していることになります．2つのレジスタ(R1)，(R2)の値が図2-2に示す約束にしたがって加算され，CPUアドレスが生成されます（図1-3参照）．

レジスタ(R1)の値，つまりセグメントアドレスを固定した場合，ここを基準にして，レジスタ(R2)で64Kバイトのアドレス空間を自由に指定できることになります．このように，任意のアドレス（セグメントアドレス）を基準にして，他方のレジスタ（R2)だけで指定できるアドレス空間をセグメントと呼びます．このことから，セグメントアドレスをベースアドレスとも呼びます．

PC-98の16ビットCPUには，セグメントアドレス(ベースアドレス）を指定するための専用レジスタが4本備えられています（表2-1参照）．

表2-1 セグメントレジスタ

記号	レジスタ名称	備　考
CS	コード・セグメント・レジスタ	主にプログラムを格納する領域を指定する．
DS	データ・セグメント・レジスタ	主にデータを格納する領域を指定する．
SS	スタック・セグメント・レジスタ	スタック領域を指定する．
ES	エクストラ・セグメント・レジスタ	データを格納する領域を指定する際の補助用

2.3 CPUアドレスの相対アドレス表記法

CPUアドレスを直接的に表記する場合には問題ありませんが，時には，セグメント内での相対アドレス（オフセットアドレス）が明確になるように表記した方が都合がよい場合があります．この場合には，セグメントアドレスとオフセットアドレスを組み合わせて，図2-3のように表記します．図2-2の場合を例にとって具体的に示します．

図2-3
相対アドレス表記法

セグメントアドレス：オフセットアドレス
(ベースアドレス)

(例)　1C80H　：　1234H

なお，図2-1の右側には，この相対アドレス表記法で表現したCPUアドレスを示しています．

3 メモリマップ

PC-98のメモリの利用状態についての解説を行います.

3.1 全体のメモリマップ

PC-98のメモリマップを図2-4に示します.

3.2 RAM領域のメモリマップ

図2-4に示したメモリマップのなかで，特に，RAM領域のメモリマップを図2-5に示します.メモリマップの様子は，システムの起動状態によって異なります. N_{88}-BASIC, N_{88}-日本語BASIC, N_{88}-DISK BASICのそれぞれで起動した場合を例にとって，メモリマップの様子を示しています.図2-5では，128Kバイト標準実装の場合を例にとり説明していますが，256Kバイト，384Kバイトについても基本的には同じです.つまり，シンボルテーブル以降の領域のサイズが大きくなるだけです.

以下では,ユーザがPC-98システムを活用していくうえで知っておくべき重要度の高い領域について，より詳細に説明します.

システム共通エリア，インタープリタ／LIOインターフェースエリア，DCB／FCB, I／Oバッファについて，順を追って説明します.

なお，割り込みベクタテーブルのメモリマップについては表1-4および表3-1で説明しています.

図2-4 【PC-9801 メモリマップ】

バンク番号	物理アドレス	用途	
	FFFFFH		
0FH	F0000H	BASIC, BIOS ROM空間	
0EH	E8000H	グラフィックVRAM	*1
	E0000H	拡張ROM空間	
0DH	D0000H		
0CH	C8000H	ユーザーROM空間	
	C0000H	グラフィックVRAM	*2
0BH	B0000H		
0AH	A8000H		
	A5000H	テキストVRAM	*3
	A0000H	メインRAM	VX, UX, VM21, UV21標準実装*4 640KB
09H	90000H		
08H	80000H		
07H	70000H		
06H	60000H		VM0, 2, 4, UV 標準実装 384KB
05H	50000H		
04H	40000H		F3, M, VF標準実装 256KB
03H	30000H		
02H	20000H		E/F1, 2/U標準実装 128KB
01H	10000H		
00H	00000H		

*1 16色表示用VRAM
VX/UX/UV/UV21/VM21は標準実装. U/VM0, 2, 4/VFはオプション. E/F/Mは実装不可能. バンク切り替えにより2プレーン有り. ただし, U用のみ1プレーン（増設不可）.

*2 Uは1プレーンのみ(増設不可). 他はバンク切り替えにより2プレーン有り.

*3 Eに漢字ROMボードを取り付けない場合, A0000H～A1FFFHは偶数アドレスのみ. A2000H～A3FFFHは全機種偶数アドレスのみ. A4000H～A5000HはVX/UXのみCGウィンドー. 他機種はA000Hからのイメージ.

【VX 80286・プロテクトモード時のみアクセス可能】

物理アドレス	用途	
FFFFFFH	0AH～0FHバンクと同じ	*5
FA0000H	予約済	
800000H	オプションRAM空間（7Mバイト）	
100000H		

*4 VX/UX/VM21/UV21ではバンク08H, 09HをDIPスイッチで切り離し可能.

*5 VX/UXの80286モードでリセット直後は初期状態のため,. リアルモードでも物理アドレスFFFFF0H番地から（初期化ルーチンが）実行される.
一旦CSレジスタを変更して初期状態から抜け出すと, リアルモードでは100000H～FFFFFFHはアクセスできなくなる.

** 漢字やANKのキャラクタジェネレータROMはメインメモリ上にはなく, I/Oポートを通して読み出すことができる.

図2-5 RAM領域のメモリマップ

(1) アドレス	(1) N_{88}-BASIC	(2) アドレス	(2) N_{88}-日本語BASIC	(3) アドレス	(3) N_{88}-DISK BASIC	サイズ (KB)
1FFFFH	機械語プログラムエリア	1FFFFH	機械語プログラムエリア	1FFFFH	機械語プログラムエリア	
	配列データエリア ↑↓		配列データエリア		配列データエリア	
	ストリングワークエリア		ストリングワークワークエリア		ストリングワークエリア	2
	ストリングデータエリア ↓		ストリングデータエリア ↓		ストリングデータエリア ↓	
		19800H	シンボルテーブル ↑			
		16000H	DISK CODE: 表示選択機能	16000H	シンボルテーブル ↑	14
		14000H	DISK CODE: DISK LIO		DISK LIO	8
		12000H	DISK CODE: N_{88}-BASIC インタープリタ		N_{88}-BASIC インタープリタ	8
	シンボルテーブル ↑↓	10000H	DISK CODE: DISK LIO		DISK LIO	8
	データスタックエリア ↓		データスタックエリア		データスタックエリア	0.5
	システムスタックエリア ↓		システムスタックエリア		システムスタックエリア	1
	プログラムテキストエリア ↑		プログラムテキストエリア		プログラムテキストエリア	
2300H	FCB I/Oバッファ	2300H	DCB/FCB I/Oバッファ	2300H	DCB/FCB I/Oバッファ	
2100H	トランスレータ出力エリア		トランスレータ出力エリア		トランスレータ出力エリア	0.5
1A00H	インタープリタ共通エリア		インタープリタ共通エリア		インタープリタ共通エリア	1.75
600H	インタープリタ/LIOインタフェースエリア		インタープリタ/LIOインタフェースエリア		インタープリタ/LIOインタフェースエリア	5
400H	システム共通エリア		システム共通エリア		システム共通エリア	0.5
000H	割り込みベクタテーブル		割り込みベクタテーブル		割り込みベクタテーブル	1

(1)N_{88}-BASIC　(2)N_{88}-日本語BASIC　(3)N_{88}-DISK BASIC　サイズ(KB)

(1)システム共通エリアのメモリマップ

システム共通エリアには，システムを構成する多数のハードウェアエレメントに関する制御情報のうち，特にエレメント相互間で知っておくと都合のよい共用性の高い内容が記録されています．ディスク制御に関するもの，描画制御に関するもの，キーボード制御に関するもの，RS-232C, GP-IB等のインターフェース制御に関するもの，等など，多種多様です．

システム共通エリアのメモリマップを**表2-2**に示します．

システム共通エリアは，細かくブロックに分けられていて，それぞれにブロック名が付けられています．**表2-2**では，各ブロックに対応するBIOSの種類も示しています．特に，DISK BIOS，キーボードBIOS，CRT BIOSに関したブロックについては，詳細を**表2-2(1)，(2)，(3)**に示しています．

表2-2 システム共通エリアのメモリマップ

ブロック名	相対アドレス*	サイズ(バイト)	解説	関連BIOS名
MS-DOS	000H	128	MS-DOSで使用	
	080H	44	未使用	
2HD-MODE	093H	1	1M/640K両用インタフェースが1Mモードの時，接続されている各ユニットに対するアクセスモードを指定する情報	DISK
DISK-EQUIP2	094H	1	1M/640K両用インタフェースが640Kモードの時，接続されている1MBドライブの接続状況を示す情報	DISK
GR-CHG	095H	1	グラフィックチャージャの制御情報	DISK
GR-TAL	096H	4	グラフィックチャージャのタイルレジスタ＃0,1,2,3の設定値	
XROM-PTR	0ACH	4	拡張ROMの初期化ルーチンが参照するポインタ	
DISK-XROM	0B0H	16	DISK BIOSが拡張ROMへアクセスする場合のポインタ	DISK
XROM-ID	0C0H	64	拡張ROMの各ロケーションの識別コードが格納される	
BIOS-FLG	100H	2	BIOS制御用フラグ	
KB-BUF	102H	32	キーコードバッファ	キーボード
KB-TBL	122H	2	キーコード変換テーブルのオフセットアドレス	キーボード
KB-HEAD	124H	2	キーコードバッファの格納済エリアの先頭オフセットアドレス	キーボード
KB-TAIL	126H	2	キーコードバッファの格納済エリアの最終アドレス＋1	キーボード
KB-COUNT	128H	1	キーコードバッファの格納済キーコード数	キーボード
KB-RTRY	129H	1	キーボードのI/O制御におけるエラーリトライの回数	キーボード
KB-STS	12AH	16	キーの押下状態を示すテーブル	キーボード
KB-SHFT	13AH	1	シフトキーの押下状態を示すフラグ	キーボード

＊セグメントアドレス＝0040H

ブロック名	相対アドレス*	サイズ(バイト)	解説	関連BIOS名
CR-RAST	13BH	1	CRTの行当りラスタ数を指定	CRT
CR-FLG	13CH	1	CRTの状態を示すフラグ	CRT
CR-CNT	13DH	1	作業用カウンタ，CRT BIOSで使用	CRT
CR-OFST	13EH	2	CRT制御パラメータブロックのオフセットアドレス	CRT
CR-SEG	140H	2	CRT制御パラメータブロックのセグメントアドレス	CRT
CR-FONT	146H	1	CGから読み出す文字フォントパターン	CRT
CR-NO	147H	1	GDCに設定する部分画面の個数	CRT
CR-VRAM	148H	2	VRAMの表示開始オフセットアドレス	CRT
CR-RAST0	14AH	2	表示画面全体のラスタ本数	CRT
CRT	14CH	1	CRTの状態を示すフラグ	CRT
G-DMODE	14DH	1	GDCに設定したドット修正モード情報	CRT
G-LPTN	14EH	2	GDCに設定した線種パターン情報	CRT
G-CPTN	14EH	8	GDCに設定したグラフィック文字パターン情報	CRT
RS-OFST	156H	2	RS-232C受信バッファのオフセットアドレス	RS-232C
RS-SEG	158H	2	RS-232C受信バッファのセグメントアドレス	RS-232C
RS-FLG	15BH	1	RS-232C受信データのシフト状態を示すフラグ	RS-232C
DISK-EQUIP	15CH	2	ディスク装置の接続状況を示す情報	DISK
DISK-INT	15EH	2	ディスク装置からの割り込み状況を示すフラグ	DISK
DISK-TYPE	160H	1	5″FDのタイプに関する情報	DISK
DISK-MODE	161H	1	5″FDのオペレーティングモードに関する情報	DISK
DISK-TIME	162H	2	5″FDのタイムアウトチェック用カウンタ	DISK
DISK-RSLT	164H	32	FDCから戻される制御情報	DISK
DISK-BOOT	184H	1	システムディスク装置のアドレス	DISK
DISK-STS	185H	1	5″HDから戻される完了時ステータス情報	
DISK-SENS	186H	4	5″HDから戻されるセンス情報	
TIM	18AH	2	インターバルタイマの設定値．タイマBIOSで使用	タイマ
DISK-WORK	18CH	2		DISK
G-PAINT	18EH	50	PAINT処理の高速制御用エリア．(グラフィックLIO・グラフィックBIOS)で使用	CRT
DISK-RST	192H	1		DISK
DIPSW	1C0H	1	DIP SWの設定状態	
RS-FLG	1C1H	1	RS-232C受信データ中のDELコードの扱いを指定する情報	RS-232C
GP-WORK	1C2H	4	GP-IBの作業領域のオフセット，セグメントアドレス．GP-IB BIOSで使用	GP-IB
KB-CODE	1C6H	4	キーボードのコード変換テーブルへのポインタ	キーボード
2DD-MODE	1CAH	1	640KB FDDに対するオペレーションモードを設定する	DISK
2DD-COUNT	1CBH	1	640KB FDDのモータOFFまでのタイマ値	DISK
2DD-POINT	1CCH	4	640KB FDDのコマンドに対応するパラメータテーブルへのポインタ	DISK
2DD-RSLT	1D0H	16	640KB FDDのI/O終了時，FDDから戻されるステータス情報	DISK
MUSIC-WORK	1E0H	4	MUSIC BIOS用ワークエリア	

* セグメントアドレスはDS＝0040H

表2-2(1) システム共通エリアメモリマップ(DISK BIOS関係)

<table>
<tr><th>ブロック名</th><th>相対 *
アドレス</th><th>サイズ
(バイト)</th><th>説　　明</th></tr>
<tr><td>2HD-MODE</td><td>093H</td><td>1</td><td>両用インターフェースが1MBモードに設定されているとき，FDD装置へのアクセスモードを指定する情報が記入されています．(初期値＝FFH)
<table>
<tr><th>ビット</th><th>ユニット名</th><th>ビット値＝0</th><th>ビット値＝1</th></tr>
<tr><td>b_0</td><td>ユニット＃0</td><td>片面モード</td><td>両面モード</td></tr>
<tr><td>b_1</td><td>ユニット＃1</td><td>片面モード</td><td>両面モード</td></tr>
<tr><td>b_2</td><td>ユニット＃2</td><td>片面モード</td><td>両面モード</td></tr>
<tr><td>b_3</td><td>ユニット＃3</td><td>片面モード</td><td>両面モード</td></tr>
<tr><td>b_4</td><td>ユニット＃0</td><td>48tpiモード</td><td>96tpi倍トラックモード</td></tr>
<tr><td>b_5</td><td>ユニット＃1</td><td>48tpiモード</td><td>96tpi倍トラックモード</td></tr>
<tr><td>b_6</td><td>ユニット＃2</td><td>48tpiモード</td><td>96tpi倍トラックモード</td></tr>
<tr><td>b_7</td><td>ユニット＃3</td><td>48tpiモード</td><td>96tpi倍トラックモード</td></tr>
</table></td></tr>
<tr><td>DISK-EQUIP2</td><td>094H</td><td>1</td><td>両用インターフェースが640KBモードに設定されているとき，接続されている1MB FDDの接続状態を示す．
<table>
<tr><th>ユニット名</th><th>DA・UA</th></tr>
<tr><td>ユニット＃0</td><td>F0H</td></tr>
<tr><td>ユニット＃1</td><td>F1H</td></tr>
<tr><td>ユニット＃2</td><td>F2H</td></tr>
<tr><td>ユニット＃3</td><td>F3H</td></tr>
</table>
DA(上位4ビット)＝デバイス番号
UA(下位4ビット)＝ユニット番号

DA・UAを合わせてデバイスアドレスと呼ぶ．</td></tr>
<tr><td>DISK-EQUIP</td><td>15CH</td><td>2</td><td>INITIALIZEコマンド実行時に，接続されている装置ユニットの状態がセットされる(ビット値＝1のとき，ユニットが接続されていることを示す)．ビット番号と装置ユニットの対応関係は，下記の通り．
<table>
<tr><td>b_0</td><td>1MB FDユニット＃0</td><td>b_8</td><td>HDユニット＃0</td></tr>
<tr><td>b_1</td><td>1MB FDユニット＃1</td><td>b_9</td><td>HDユニット＃1</td></tr>
<tr><td>b_2</td><td>1MB FDユニット＃2</td><td>b_A</td><td>――――</td></tr>
<tr><td>b_3</td><td>1MB FDユニット＃3</td><td>b_B</td><td>――――</td></tr>
<tr><td>b_4</td><td>320KB ユニット＃0</td><td>b_C</td><td>640KB FDユニット＃0</td></tr>
<tr><td>b_5</td><td>320KB ユニット＃1</td><td>b_D</td><td>640KB FDユニット＃1</td></tr>
<tr><td>b_6</td><td>320KB ユニット＃2</td><td>b_E</td><td>640KB FDユニット＃2</td></tr>
<tr><td>b_7</td><td>320KB ユニット＃3</td><td>b_F</td><td>640KB FDユニット＃3</td></tr>
</table></td></tr>
</table>

＊セグメントアドレスはDS＝0040H

<table>
<tr><th>ブロック名</th><th>相対
アドレス</th><th>サイズ
(バイト)</th><th>説　　　明</th></tr>
<tr><td>DISK-INT</td><td>15EH</td><td>2</td><td>デバイスからの割り込み情報（ビット値＝1のとき，対応するデバイスから割り込みがあったことを示す）
ビット番号とデバイスの対応関係は下記の通り．
<table>
<tr><td>b_0</td><td>1MB FDユニット＃0</td><td>b_8</td><td>5″HDユニット＃0</td></tr>
<tr><td>b_1</td><td>1MB FDユニット＃1</td><td>b_9</td><td></td></tr>
<tr><td>b_2</td><td>1MB FDユニット＃2</td><td>b_{10}</td><td></td></tr>
<tr><td>b_3</td><td>1MB FDユニット＃3</td><td>b_{11}</td><td></td></tr>
<tr><td>b_4</td><td></td><td>b_{12}</td><td>640KB FDユニット＃0</td></tr>
<tr><td>b_5</td><td></td><td>b_{13}</td><td>640KB FDユニット＃1</td></tr>
<tr><td>b_6</td><td></td><td>b_{14}</td><td>640KB FDユニット＃2</td></tr>
<tr><td>b_7</td><td></td><td>b_{15}</td><td>640KB FDユニット＃3</td></tr>
</table></td></tr>
<tr><td>DISK-TYPE</td><td>160H</td><td>1</td><td>接続されている5″FDのタイプ（PC-9801/E/F/Mのみ）
<table>
<tr><td>FFH</td><td>片面タイプ</td></tr>
<tr><td>EFH</td><td>両面タイプ</td></tr>
<tr><td>00H</td><td>無</td></tr>
</table></td></tr>
<tr><td>DISK-MODE</td><td>161H</td><td>1</td><td>接続されている両面タイプの5″FDに対するオペレーティングモード
(PC-9801/E/F/Mのみ)
<table>
<tr><th>ビット</th><th>ユニット名</th><th>ビット値＝0</th><th>ビット値＝1</th></tr>
<tr><td>b_0</td><td>ユニット＃0</td><td>片面アクセス</td><td>両面アクセス</td></tr>
<tr><td>b_1</td><td>ユニット＃0</td><td>片面アクセス</td><td>両面アクセス</td></tr>
<tr><td>b_2</td><td>ユニット＃0</td><td>片面アクセス</td><td>両面アクセス</td></tr>
<tr><td>b_3</td><td>ユニット＃0</td><td>片面アクセス</td><td>両面アクセス</td></tr>
</table></td></tr>
<tr><td>DISK-TIME</td><td>162H</td><td>2</td><td>5″FDが入出力が完了するまでの待ち時間を設定する．
(PC-9801/E/F/Mのみ)
<table>
<tr><td>0000H</td><td>完了するまで無条件に待つ</td></tr>
<tr><td>nH</td><td>n（msec）</td></tr>
</table></td></tr>
</table>

<table>
<tr><th>ブロック名</th><th>相対
アドレス</th><th>サイズ
(バイト)</th><th>説　　明</th></tr>
<tr><td>DISK-RSLT</td><td>164H</td><td>32</td><td>8"FDに対するFDCから戻されるリザルトステータス情報が格納される. 1ユニットに対し, 8バイトが割り付けられていて, 4ユニット分ある. 他のユニットについても同様.
<table>
<tr><td>B+0</td><td>ST0(リザルトステータス)</td><td></td></tr>
<tr><td>B+1</td><td>ST1(リザルトステータス)</td><td></td></tr>
<tr><td>B+2</td><td>ST2(リザルトステータス)</td><td></td></tr>
<tr><td>B+3</td><td>C シリンダ番号</td><td rowspan="3">物理アドレス</td></tr>
<tr><td>B+4</td><td>H ヘッド番号</td></tr>
<tr><td>B+5</td><td>R レコード番号</td></tr>
<tr><td>B+6</td><td>N レコード内のデータ長</td><td></td></tr>
<tr><td>B+7</td><td>現在のシリンダ番号</td><td></td></tr>
</table></td></tr>
<tr><td>DISK-BOOT</td><td>184H</td><td>1</td><td>システムディスク装置のデバイスアドレス
上位4ビット＝DA(デバイス番号)
下位4ビット＝UA(ユニット番号)
<table>
<tr><th>DA</th><th>種別</th></tr>
<tr><td>5H</td><td>5"FD 2D</td></tr>
<tr><td>7H</td><td>5"FD 2DD</td></tr>
<tr><td>8H</td><td>5"HD</td></tr>
<tr><td>9H</td><td>8"FD 2D</td></tr>
</table></td></tr>
<tr><td>DISK-RESET</td><td>192</td><td>2</td><td>リキャリブレイトすべきユニットを示す. (ビット値＝1の時, 対応するユニットに対し, リキャリブレイトが実行される
<table>
<tr><td>b0</td><td>1MB FDユニット＃0</td><td>b5</td><td>640KB FDユニット＃0</td></tr>
<tr><td>b1</td><td>1MB FDユニット＃1</td><td>b5</td><td>640KB FDユニット＃1</td></tr>
<tr><td>b2</td><td>1MB FDユニット＃2</td><td>b6</td><td>640KB FDユニット＃2</td></tr>
<tr><td>b3</td><td>1MB FDユニット＃3</td><td>b7</td><td>640KB FDユニット＃3</td></tr>
</table></td></tr>
<tr><td>2DD-MODE</td><td>1CAH</td><td>1</td><td>640KB FDユニットに対するアクセスモードを指定(初期値＝FFH)
<table>
<tr><th>ビット</th><th>ユニット名</th><th>ビット値＝0</th><th>ビット値＝1</th></tr>
<tr><td>b0</td><td>ユニット＃0</td><td>片面モード</td><td>両面モード</td></tr>
<tr><td>b1</td><td>ユニット＃1</td><td>片面モード</td><td>両面モード</td></tr>
<tr><td>b2</td><td>ユニット＃2</td><td>片面モード</td><td>両面モード</td></tr>
<tr><td>b3</td><td>ユニット＃3</td><td>片面モード</td><td>両面モード</td></tr>
<tr><td>b4</td><td>ユニット＃0</td><td>48tpi モード</td><td>96tpi倍トラックモード</td></tr>
<tr><td>b5</td><td>ユニット＃1</td><td>48tpi モード</td><td>96tpi倍トラックモード</td></tr>
<tr><td>b6</td><td>ユニット＃2</td><td>48tpi モード</td><td>96tpi倍トラックモード</td></tr>
<tr><td>b7</td><td>ユニット＃3</td><td>48tpi モード</td><td>96tpi倍トラックモード</td></tr>
</table></td></tr>
<tr><td>2DD-COUN</td><td>1CBH</td><td>1</td><td>640KB FDのモータをOFFにするまでの時間を設定するカウント値
(単位は100msec)</td></tr>
</table>

表2-2(2) システム共通エリアのメモリマップ(キーボードBIOS関係)

ブロック名	相　対アドレス*	サイズ(バイト)	解　説
KB_BUF	102H	32	キーボードから送られる入力キーコード(2バイト)を，最大16個まで格納できるバッファ
KB_HEAD	124H	2	キーコードバッファ内の先頭キーコードの格納アドレス(オフセット)
KB_TAIL	126H	2	キーコードバッファの未使用エリアの先頭アドレス(オフセット)
KB_COUNT	128H	1	キーコードバッファに格納されているキーコードの個数
KB_STS	12AH	16	16バイト(96ビット)の各ビットが，96個のキーの押下状態を示す．キー押下時に，対応するビット値＝1
KB_SHFT	13AH	1	シフトキーの押下状態を示すフラグ．キー押下時に，対応するビット値＝1 b_0 SHIFT b_1 CAPS b_2 カナ b_3 GRPH b_4 CTRL

表2-2(3) システム共通エリアメモリマップ(CRT BIOS関係)

ブロック名	相対アドレス	サイズ(バイト)	解説
CR_RAST	13BH	1	CRTの行当りのラスタ本数を指定する. ここに代入する値は, (ラスタ本数)－1
CR_FLG	13CH	1	CRTの状態を示すフラグ

ビット	フラグ名	ビット値＝0	ビット値＝1
b_0	ラインモード	25行	20行
b_1	カラムモード	80カラム	40カラム
b_2	アトリビュート	垂線表示	簡易グラフ
b_3	K-CGモード	コードアクセス	ドットアクセス
b_7	CRTタイプ	標準CRT	高解像CRT

ブロック名	相対アドレス	サイズ(バイト)	解説
CR_FONT	146H	1	CGから読み出す文字フォントパターンの種別を示す.

ビット	名　称	ビット値＝0	ビット値＝1
b_0	フォントサイズ	6×7	7×11
b_1	タイプ	ANK文字	漢字

ブロック名	相対アドレス	サイズ(バイト)	解説
CRT	14CH	1	グラフィックCRTの状態を示すグラフ

ビット	名　称	ビット値＝0	ビット値＝1
b_0	BASICモード	互換	拡張
b_1	グラフィックチャージャ	無し	有り
b_2	G-VRAM(拡張)	無し	有り
b_3	ユーザ定義文字	63文字	188文字
b_4			
b_5			
b_6	CRTタイプ	標準CRT	高解像CRT
b_7	CRT状態	表示停止	表示

ブロック名	相対アドレス	サイズ(バイト)	解説
G_DMODE	14DH	1	GDCに設定したドット修正モード情報

b_1	b_0	ドット修正モードの設定状態
0	0	REPLACE
0	1	COMPLEMENT
1	0	CLEAR
1	1	SET

⑵インタープリタ／LIO　インターフェースエリア

BASICインタープリタとLIO(GRAPH LIO, DISK LIO etc)とのインターフェースのための領域について説明します．このインターフェースエリアのメモリマップを**図2-6**に示します．

このなかで，特にDISK UCW*を取り上げて，より詳細なメモリマップを表**2-3**に示します．なお，DISK UCWとは，ディスク装置の制御情報が記録されている領域のことです．

図2-6　インタープリタ／LIO　インターフェースエリアのメモリマップ

セグメントアドレス　オフセットアドレス		サイズ（バイト）
0060H：13FFH	COPY ワークエリア	0.5K
	LIO ワークエリア	0.5K
0060H：0A00H	インタープリタ コンスタントエリア	1.5K
	ターミナル	76
	RS-232C　UCW	532
0060H：06A0H	インタープリタ共通エリア	256
	グラフィック UCW	128
0060H：0500H	DISK/PR　UCW	288
0060H：0000H	キーボード/CRT　UCW	1.28K

＊UCW＝Unit Control Work

表2-3 DISK UCWのメモリマップ

フィールド名称*	相対アドレス**	サイズ(バイト)	説明
UC_5FD	501H	1	5"FD装置の数　(最大4台)
UC_8FD	502H	1	1MB FD装置の数　(最大4台)
UC_DSK	503H	1	ディスク装置の合計数(最大12台)
UC_DOPN	504H	1	同時にOPENするファイルの数(00H～0FH)
UC_SRVT	505H	1	SRVの種別 01H：N_{88}-BASIC 02H：N-BASIC(5"FD 1D) 03H：N-BASIC(5"FD 2D)
UC_DBUF	506H	2	ディスクのPIOバッファの先頭アドレス
UC_DDCB	508H	2	ディスクのDCB群の先頭アドレス
UC_DFCB	50AH	2	ディスクのFCB群の先頭アドレス
UC_FATB	50CH	2	FATバッファの先頭アドレス
UC_DCON	50EH	2	媒体の諸元格納テーブルの先頭アドレス
UC_USID	510H	3	ユーザ識別子 090909H：システム用 その他　：ユーザ用

(注) ただし，FDDに関するもののみ示しており，HDに関するものは除外している．

* フィールド名称のUC-はUCW内のフィールドであることを明記するためにつけている添字である．

** ベースアドレス（セグメントアドレス）＝0060H

第2章 メモリ

⑶PICB, DCB, FCBのメモリマップ

PICB, DCB, FCB*のメモリマップを図2－7に示します．これらは，ディスク装置に対するデータの入出力制御に関する領域です．

次に，PICB, DCB, FCBのより詳細なメモリマップをそれぞれ表2-4，表2-5，表2-6に示します．

図2-7 DCB/FCB，PICB(min644B～max15.724KB)

アドレス	内容	サイズ(バイト)
	PIOバッファ	256×(ファイルオープン数＋1)
	FCB	40×(ファイルオープン数＋1)
	PICB	24×装置タイプ数
	FATバッファ	256×装置台数(注)
：1D90H	DCB	20×装置台数
0060H：1D00H	媒体諸元表	48×装置タイプ数
(ベースアドレス)(オフセットアドレス)		

(注) ただし，FDDの場合のみ

* PICB＝Physical Input output Control Block
DCB＝Device Control Block
FCB＝File Control Block

表2-4 PICBのメモリマップ

<table>
<tr><th>フィールド名</th><th>相　対
アドレス*</th><th>サイズ
(バイト)</th><th>説　　明</th></tr>
<tr><td>PI-IOS</td><td>00H</td><td>1</td><td>I/Oステータス
<table>
<tr><th></th><th>SENSEコマンド時</th><th>SENSEコマンド以外</th></tr>
<tr><td>b_0</td><td>———</td><td>MISSING ADDRESS MARK</td></tr>
<tr><td>b_1</td><td>———</td><td>書き込み禁止</td></tr>
<tr><td>b_2</td><td>———</td><td>NO DATA</td></tr>
<tr><td>b_3</td><td>両面FDD</td><td>NOT READY</td></tr>
<tr><td>b_4</td><td>TRACK 0</td><td>OVER RUN</td></tr>
<tr><td>b_5</td><td>READY</td><td>DATA ERROR</td></tr>
<tr><td>b_6</td><td>ライトプロテクト</td><td>DEVICE CHECK</td></tr>
<tr><td>b_7</td><td>———</td><td>END OF TRACK</td></tr>
</table>
(注) ビット値＝1の時，上記状態の発生を示す．</td></tr>
<tr><td>PI-CMD</td><td>01H</td><td>1</td><td>DISK BIOSルーチンに対するコマンドコード
<table>
<tr><td>01H：</td><td>VERIFY</td></tr>
<tr><td>03H：</td><td>INITIALIZE</td></tr>
<tr><td>04H：</td><td>SENSE</td></tr>
<tr><td>05H：</td><td>WRITE DATA</td></tr>
<tr><td>06H：</td><td>READ DATA</td></tr>
<tr><td>07H：</td><td>RECALIBRATE</td></tr>
<tr><td>09H：</td><td>WRITE DELETED DATA</td></tr>
<tr><td>0AH：</td><td>READ ID</td></tr>
<tr><td>0DH：</td><td>WRITE ID</td></tr>
<tr><td>0FH：</td><td>SEEK</td></tr>
</table></td></tr>
<tr><td>PI-DTA</td><td>02H</td><td>2</td><td>I/Oの対象となるデータの先頭アドレス(オフセットアドレス)
(注) 偶数アドレスでなければならない</td></tr>
<tr><td>PI-DTS</td><td>04H</td><td>2</td><td>I/Oの対象となるデータの先頭アドレス(ベースアドレス)</td></tr>
<tr><td>PI-DTL</td><td>06H</td><td>2</td><td>I/Oの対象となるデータの長さ［単位：バイト］</td></tr>
<tr><td>PI-DCF</td><td>08H</td><td>6</td><td>指定セクタの物理アドレス
<table>
<tr><td>第1バイト</td><td>—</td></tr>
<tr><td>第2バイト</td><td>—</td></tr>
<tr><td>第3バイト</td><td>H (ヘッド番号)</td></tr>
<tr><td>第4バイト</td><td>R (セクタ番号)</td></tr>
<tr><td>第5バイト</td><td rowspan="2">C (シリンダ番号)</td></tr>
<tr><td>第6バイト</td></tr>
</table></td></tr>
</table>

フィールド名	相　対 アドレス	サイズ (バイト)	説　　明
PI-CLST	0EH	2	I/O の対象となるクラスタ番号
PI-BLK	10H	1	I/O を実行すべきブロック番号を，クラスタ内の相対値で示す
PI-RFU1	11	1	予約域
PI-TIME	12H	2	I/O 時のタイム・アウト処理時間
			{ 0800H : N-BASIC
			2800H : N_{88}-BASIC
PI-RFU2	14H	4	予約域

(注1)ディスク装置のタイプ数分だけ，これと同じ24バイトのPICBが作成される．
図5-5参照

(注2) PICBの先頭アドレスはDCB上のフィールドDC-PICBに格納されている．ただし，ベースアドレス＝0060Hである．

(注3) フィールド名のPI-はPICB内のフィールドであることを明記するためにつけている添字である．

表2-5　DCBのメモリマップ

<table>
<tr><th>フィールド名</th><th>相　対
アドレス*</th><th>サイズ
(バイト)</th><th>説　明</th></tr>
<tr><td>DC-DRNO</td><td>00H</td><td>1</td><td>ドライブ番号(01H～0EH)</td></tr>
<tr><td>DC-DVAD</td><td>01H</td><td>1</td><td>物理デバイスアドレス　b_7 b_6 b_5 b_4 b_3 b_2 b_1 b_0
デバイスアドレス
b_7～b_4＝ { 1001：1MBFDD
1000：5″ハードディスク
0111：5″FDD(2DD) }
ユニットアドレス
b_3～b_0＝ { 0000：ユニット#1
0001：ユニット#2
0010：ユニット#3
0011：ユニット#4 }</td></tr>
<tr><td>DC-FCB</td><td>02H</td><td>2</td><td>デバイスに属する最初のファイルに関するFCBの先頭アドレス
(注) オープンされたファイルがない時は，0000H</td></tr>
<tr><td>DC-DSTS</td><td>04H</td><td>1</td><td>デバイス，ステータス　b_7 b_6 b_5 b_4 b_3 b_2 b_1 b_0
<table>
<tr><th></th><th></th><th>ビット値＝0</th><th>ビット値＝1</th></tr>
<tr><td>b_0</td><td>マウント状態</td><td>未</td><td>既</td></tr>
<tr><td>b_1</td><td>5″FDのモード</td><td>2D 可</td><td>2D 不可</td></tr>
<tr><td>b_2</td><td>媒体種別</td><td>片面FDDまたは
5″ハードディスク1台目</td><td>両面FDDまたは
5″ハードディスク2台目</td></tr>
<tr><td>b_3</td><td>FAT更新</td><td>未更新</td><td>更新</td></tr>
<tr><td>b_4</td><td>5″ハードディスク
マウント状態</td><td>ボリュームラベル
未READ</td><td>READ済</td></tr>
<tr><td>b_6</td><td>FAT書換えID</td><td colspan="2">FATテーブル更新毎にON, OFFし，OFFの時，FATを更新する
(N_{88}-BASICのみ)</td></tr>
<tr><td>b_7</td><td>削除ディレクトリID
(WORKとして使用)</td><td>有り</td><td>無し</td></tr>
</table></td></tr>
<tr><td>DC-ID</td><td>05H</td><td>1</td><td>ボリュームの属性を示す
b_0～b_3　MB2
b_4 ― 0：書込可，1：禁止
b_5 ― MB2
b_6 ― 0：Write，1：Read after Write
b_7 ― ME2</td></tr>
</table>

表2-5　DCBのメモリマップ

<table>
<tr><th>フィールド名</th><th>相　対
アドレス*</th><th>サイズ
(バイト)</th><th>説　明</th></tr>
<tr><td>DC-FAT</td><td>06H</td><td>2</td><td>FATバッファの先頭アドレス</td></tr>
<tr><td>DC-PICB</td><td>08H</td><td>2</td><td>PICBの先頭アドレス</td></tr>
<tr><td>DC-USCL</td><td>0AH</td><td>2</td><td>ボリューム内の未使用クラスタ数</td></tr>
<tr><td>DC-DIRS</td><td>0CH</td><td>4</td><td>カレント・ディレクトリの物理アドレス
<table>
<tr><td>第1バイト</td><td>サーフェス番号</td></tr>
<tr><td>第2バイト</td><td>セクタ番号</td></tr>
<tr><td>第3バイト</td><td rowspan="2">トラック番号</td></tr>
<tr><td>第4バイト</td></tr>
</table></td></tr>
<tr><td>DC-D1RP</td><td>10H</td><td>1</td><td>カレントディレクトリのセクタ内の相対位置</td></tr>
<tr><td>DC-DAUA</td><td>11H</td><td>1</td><td>物理デバイスアドレス(ユニットアドレスとデバイスアドレスから構成される)
<table>
<tr><th>ビット番号</th><th></th></tr>
<tr><td>b_3 b_2 b_1 b_0</td><td>ユニットアドレス　0000：ユニット#1
0001：ユニット#2
0010：ユニット#3
0011：ユニット#4</td></tr>
<tr><td>b_7 b_6 b_5 b_4</td><td>デバイスアドレス
1001：1Mインターフェース，1Mアクセス
0001：1Mインターフェース，640Kアクセス
1111：640Kインターフェース，1Mアクセス
0111：640Kインターフェース, 640Kアクセス
1000：5″HD</td></tr>
</table></td></tr>
<tr><td>DC-DVTP</td><td>BH</td><td>1</td><td>b_0　0：FATを分割しない
　　1：する
b_4　0：従来ドライブ
　　1：両用</td></tr>
</table>

(注) ディスク装置の台数分だけ，これと同じ20バイトの BCD が作成される．
図5-5参照

* DCB の先頭アドレスは，DISK UCW上のフィールド UC-DDCBに格納されている．
ただし，セグメントアドレス＝0060Hである．

表2-6 FCBのメモリマップ

フィールド名	相対アドレス*	サイズ(バイト)	説明
FC_FNO	00H	1	ファイル番号(00H～0FH)
FC_OPNM	01H	1	オープンモード 80H: INPUT モード 40H: OUTPUT モード 41H: APPENDモード C0H: 指定なし
FC_DCB	02H	2	当ファイルが属するデバイスの DCB の先頭アドレス
FC_NXFC	04H	2	当ファイルが属するデバイスに属している次のファイルに関する FCB の先頭アドレス (注) オープンされている次のファイルがない時, 0000H
FC_FID	06H	6	ファイル名
FC_EID	0CH	3	ファイルの拡張子
FC_ATTR	0FH	1	ファイルの属性 b_7 b_6 b_5 b_4 b_3 b_2 b_1 b_0

ビット	ビット値＝0	ビット値＝1
b_7	ASCII形式	非ASCII形式
b_6	Write only	Read after Write
b_5		Pオプション
b_4	書き込み可	書き込み禁止
b_0		機械語形式

フィールド名	相対アドレス*	サイズ(バイト)	説明
FC-FCLS	10H	2	当ファイルの先頭クラスタ番号
FC-ATRW	12H	1	ファイルの属性 (注) ATTRと同じ．ただし，書き込み禁止のチェックは当フィールドを参照することによりなされる．

(注) 同時オープンファイルの数の指定値＋1に相当する個数だけ，これを同じ40バイトのFCBが作成される（図5-5参照）．

＊ FBCの先頭アドレスはDCB上のフィールドDC FCBに格納されている．ただし，セグメントアドレス＝0060H

<table>
<tr><th>フィールド名</th><th>相対
アドレス</th><th>サイズ
(バイト)</th><th>説明</th></tr>
<tr><td>FC-DVTP</td><td>13H</td><td>1</td><td>当ファイルが属するデバイスの種類
B0H　ディスク
B2H　プリンタ
1AH　CMT
19H　RS-232C</td></tr>
<tr><td>FC-FSTS</td><td>14H</td><td>1</td><td>ファイルの処理状態 $b_7 \sim b_0$
<table><tr><th>ビット</th><th>ビット値＝0</th><th>ビット値＝1</th></tr><tr><td>b_7</td><td>ファイルの途中</td><td>AT END 検出</td></tr><tr><td>b_1</td><td>バッファ書き戻し不要</td><td>必要</td></tr><tr><td>b_0</td><td>未オープン</td><td>オープン中</td></tr></table></td></tr>
<tr><td>FC-EOD</td><td>15H</td><td>3</td><td>最終レコードアドレス { 上位2バイト＝クラスタ番号
下位1バイト＝ブロック番号 }</td></tr>
<tr><td>FC-LRNO</td><td>18H</td><td>2</td><td>最終レコード番号</td></tr>
<tr><td>FC-NAD</td><td>1AH</td><td>3</td><td>次レコードアドレス { 上位2バイト＝クラスタ番号
下位1バイト＝ブロック番号 }</td></tr>
<tr><td>FC-NRNO</td><td>1DH</td><td>2</td><td>次レコード番号</td></tr>
<tr><td>FC-RFU1</td><td>1FH</td><td>1</td><td>(予備)</td></tr>
<tr><td>FC-PBUF</td><td>20H</td><td>2</td><td>当FCB に対応する PIO バッファの先頭アドレス</td></tr>
<tr><td>FC-RFU1</td><td>22H</td><td>2</td><td>(予備)</td></tr>
<tr><td>FC-OFST</td><td>24H</td><td>2</td><td>次レコード空間のオフセット</td></tr>
<tr><td>FC INTU</td><td>26H</td><td>2</td><td>(インタープリタが使用)</td></tr>
</table>

第3章

内部ルーチンの活用

CONTENT

1 概要

PC-98のROM上には，N_{88}-BASICの強力な機能の背景となっている完成度の高いすぐれたルーチンが多数あります．このROM上のルーチンが活用できれば，ユーザのアプリケーションプログラム開発の効率が格段に向上します．ただし，ROMのバージョンによっては，各ルーチンの先頭アドレスの割り付けが変更されているためユーザにとって頭の痛い問題となっています（図2-5参照）．しかし，幸いなことに，PC-98では割り込みベクタテーブルという領域が用意されていて，ここにそれぞれのルーチンの先頭アドレスが格納されています．各ルーチンには，ベクタコードが割り当てられていて，ユーザは目的とするルーチンに対応するベクタコードさえ知っていれば，容易にルーチンへアクセスできるのです．

このように，ベクタコードに基づいて目的とするルーチンをコールする手続きのことをソフトウェア割り込み*と呼んでいますが，本章では，このソフトウェア割り込みを用いて，ROM上のルーチンを活用するための解説を行います．

＊ インターラプトコールとも呼びます．

2 ソフトウェア構造

PC-98のシステムのソフトウェア，つまりN_{88}-BASICの構造の概要を**図3-1**に示します．PC-98のソフトウェアは，BIOS, LIO*, BASICインタープリタの3階層に分類して考えることができます．

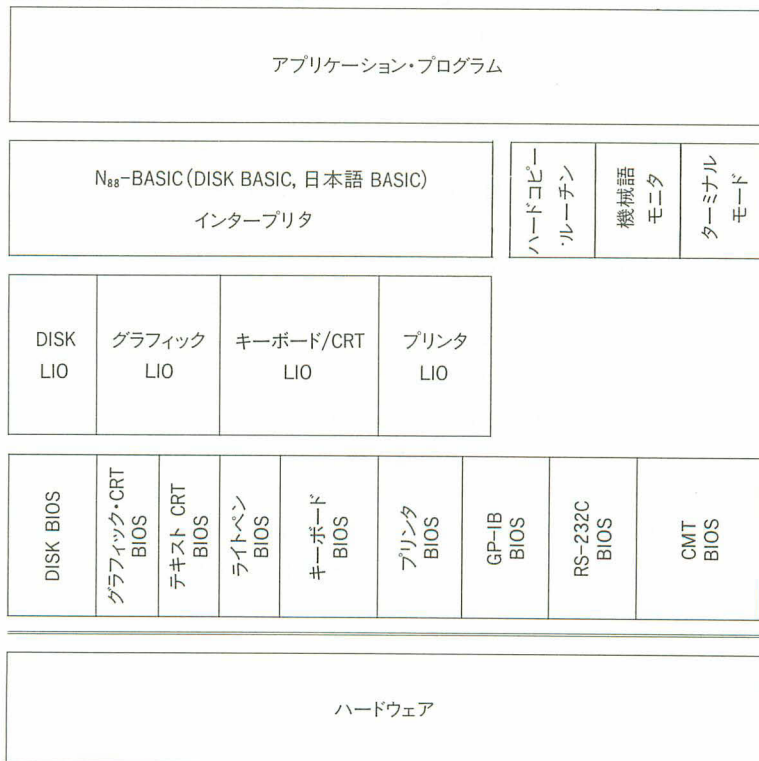
図3-1 N_{88}-BASIC のソフトウェア

アプリケーション・プログラム

N_{88}-BASIC(DISK BASIC, 日本語 BASIC) インタープリタ | ハードコピー・ルーチン | 機械語モニタ | ターミナルモード

DISK LIO | グラフィック LIO | キーボード/CRT LIO | プリンタ LIO

DISK BIOS | グラフィック・CRT BIOS | テキスト CRT BIOS | ライトペン BIOS | キーボード BIOS | プリンタ BIOS | GP-IB BIOS | RS-232C BIOS | CMT BIOS

ハードウェア

＊LIO＝Logical Input Output

3 ソフトウェア割り込み

3.1 割り込みベクタテーブル

N_{88}-BASICがBIOS, LIO, インタープリタの3階層から構成されていることについては，すでに前項で説明しました．

N_{88}-BASICを構成する多数の機能ルーチンを有効に活用するために，PC-98ではソフトウェア割り込みを用いてルーチンをコールするという方式を採用しています．

N_{88}-BASICの各ルーチンの先頭アドレスは，バージョンの違いによって異なってきており，これはユーザがルーチンを利用する場合の大きな障害となってしまいます．そこで，この問題を解消するために，PC-98では割り込みベクタテーブルという概念を採用しています．

N_{88}-BASICを構成する個々の機能ルーチンには，ベクタコードが割り当てられています．一方，割り込みベクタテーブルは4バイトを1単位として構成されており，個々の単位領域に対しても同じくベクタコードが割り当てられています．そして，ベクタコードnに対応する割り込みベクタテーブルの4バイト領域には，ベクタコードnに対応する機能ルーチンの先頭アドレス（オフセットアドレスとベースアドレス）が格納されています．したがって，ユーザは機能ルーチンに対応するベクタコードを指定すれば，割り込みベクタテーブルを介して目的とする機能ルーチンの先頭アドレスを獲得することができます．

割り込みベクタテーブルのメモリマップを**表3-1**に示します．表では，ベクタコードと，それに対応する割り込みベクタテーブル上のアドレスの対応関係を示しています．

表3-1 割り込みベクタテーブル

ベクタアドレス	ベクタコード	処理ルーチンの機能	備　　考
060H	18H	キーボード＆ CRT BIOS	第1章6.3参照／第4章6.1参照
064H	19H	RS-232C BIOS	第6章2.3参照
068H	1AH	カセット＆プリンタ BIOS	第6章5.2参照
06CH	1BH	DISK BIOS	第5章3参照
070H	1CH	カレンダ・タイマ BIOS	第1章7.2，第1章8.2参照
074H	1DH	システム予約	
078H	1EH	N_{88}-BASIC(86)のコールドスタート	
07CH 〜 0FFH	1FH 〜 3FH	システム予約	
100H 〜 1FFH	40H 〜 7FH	ユーザ定義領域	
200H	80H	キーボード＆ CRT LIO の初期化	
204H	81H	処理なし	
208H	82H	WIDTH 文の処理	
20CH	83H	キーボードからの割り込みのセンス	
210H	84H	INPUT 文の処理	N_{88}-DISK(日本語)
214H	85H	INPUT WAIT 文の処理	N_{88}-DISK(日本語)
218H	86H	キーインライン処理	N_{88}-DISK(日本語)
21CH	87H	INPUT $ 文の処理	
220H	88H	INKEY $ 文の処理	
224H	89H	PRINT 文の処理	
228H	8AH	BEEP 文の処理	
22CH	8BH	画面スクロールの処理	
230H	8CH	リターンキーの処理	
234H	8DH	PSET/PRESET 文の処理	N-BASIC
238H	8EH	POINT 文の処理	N-BASIC
23CH	8FH	GET@文の処理	N-BASIC
240H	90H	GET@A 文の処理	N-BASIC
244H	91H	PUT@文の処理	N-BASIC
248H	92H	PUT@A 文の処理	N-BASIC
24CH	93H	BOX 文の処理	N-BASIC
250H	94H	ライン・アトリビュートの設定	N-BASIC
254H	95H	COLOR@ $(x_1, y_1)-(x_2, y_2)$ 文の処理	
258H	96H	ON TIME GOSUB 文の処理	
25CH	97H	KINPUT 文の処理	N_{88}-DISK(日本語)
260H	98H	テキスト画面クリア	
264H	99H	文字コードの表示	
268H	9AH	ライトペン入力の処理	
26CH	9BH	次の物理行へカーソル移動	
270H	9CH	物理行クリア	
274H	9DH	ファンクションキーの表示	

先頭の CPU アドレス	ベクタ コード	処理ルーチンの機能	備考
278H	9EH	LIO/BIOS のキーインバッファクリア	
27CH	9FH	システム予約	
280H 〜 2BCH	A0H 〜 AFH	グラフィック LIO	第4章7参照
2C0H	B0H	DISK LIO	第5章4参照
2C4H 〜 2CCH	B1H 〜 B3H	システム予約	
2D0H	B4H	DISK LIO の初期化	第5章4参照
2D4H 〜 2FFH	B5H 〜 BFH	システム予約	
300H	C0H	ハードコピー処理	
304H	C1H	コード変換(外部コード→内部コード)	
308H	C2H	コード変換(内部コード→外部コード)	
30CH	C3H	CALL 文, USR 文の処理	
310H	C4H	BASIC インタープリタサブルーチン	第3章4参照
314H	C5H	グラフィック LIO から割り込みをセンスするためのエントリ	
318H	C6H	DISK BASIC の起動	
31CH	C7H	DISK 版エディット 機能	
320H 〜 328H	C8H 〜 CAH	BASICシステム予約	
32CH	CBH	ターミナルモード処理	
330H	CCH	リモートBASICプロトコルによるBASICステートメントの実行結果を回線へ送信する	
334H	CDH	リモートBASIC プロトコル処理	
338H	CEH	ハードコピーのグラフィック画面を読み出す	第4章7参照
33CH	CFH	機械語モニタ	
340H	D0H	機械語モニタ	
344H	D1H	GP-IB BIOS 起動	第6章3.2参照
348H	D2H	MUSIC	
34CH	D3H	BRANCH4670	
350H	D4H	第2回線 RS-232C	
354H	D5H	第3回線 RS-232C	
358H 〜 3C0H	D6H 〜 F0H	BASICシステム予約	
3C4H 〜 3FFH	F1H 〜 FFH	ユーザ定義	

3.2 ソフトウェア割り込みの手続き

PC-98では,割り込みベクタテーブルが用意されているので,ユーザはベクタコードを指定すれば,目的とする機能ルーチンをコール（ソフトウェア割り込み*）することができます.

以下では,ソフトウェア割り込みの手続きについて説明しますが,実に簡単です.

例えば,ベクタコードnに対応する機能ルーチンをコールする場合には,下記のコマンドを実行するだけでよいのです.

INT　n

ただし,ルーチンによっては,あらかじめ特定のレジスタやメモリの特定領域に,パラメータ値を設定しておく必要があります.これはちょうど高級言語におけるサブルーチンコールにともなう引数に対応しているといえます.そして,ベクタコードnがサブルーチン名に相当しているわけです.

---

* ソフトウェア割り込みのことをインターラプトコールとも呼びます.

4 BASICインタープリタ活用の手続き

ソフトウェア割り込みを用いて，活用できるルーチンの一群としてBASICインタープリタがあります．ここでは，BASICインタープリタを活用するための手続きについて説明します．

BASICインタープリタのベクタコードは，**図3-1**に示したようにC4Hです．また，BASICインタープリタ自体が多数の機能ルーチンから構成されており，個々のルーチンに対しては，**表3-2**に示すようにコマンドコードが割り当てられています．

例えば，BASICインタープリタの中で，コマンドコードmに対応する機能ルーチンをコールする場合の手続きは，以下に示す通りです．

①レジスタDIにコマンドコードmを設定する．

　DI←m

②下記のセグメントレジスタを設定する．

　DS＝60H（データのセグメントアドレス）

　SS＝60H（データのセグメントアドレス）

③必要ならば所定のレジスタおよび必要なメモリ領域にパラメータ値を設定する．

④ソフトウェア割り込みの実行．

　INT　C4H

表3-2 BASIC インタープリタ(ROM内)サブルーチン

コマンドコード	機　　能
00H	エラーメッセージの表示　　入力設定：AL←エラーコード
01H	Syntax error を表示
02H	Illegal function call を表示
03H	Type mismatch を表示
04H	エラーメッセージを表示の後，処理続行
05H	倍精度加算　FAC←FAC1＋FAC*
06H	倍精度乗算　FAC←FAC1×FAC*
07H	倍精度除算　FAC←FAC1/FAC*
08H	倍精度の整数化　FAC←INT (FAC)*
09H	倍精度化　FAC←CDBL(FAC)*
0AH	オーバーフロー処理
0BH	ゼロ除算処理
0CH	数値の整数化　FAC←CINT(FAC)*
0DH	トークン中間コード抽出 {入力：[6EAH]←テキストアドレス** 出力：SI←次のトークンのアドレス BL←解析したトークン}
0EH	数式評価 {入力：[6EAH]←評価するテキストアドレス 出力：FAC←演算結果*}
0FH	ストリングデータエリアの通知
10H	テキストの表現変換 {入力：内部表現のテキストアドレス 出力：外部表現の出力先バッファアドレス}
11H	行番号のバイナリ化　　入力：SI←文字列の先頭アドレス
12H	ファイル番号，FCB アドレスのチェック {入力：[6EAH]←ファイル番号部分を示すアドレス 出力：[1536H]←ファイル番号 [1538H]←FCB アドレス}
13H	ファイルディスクリプタの解析 {入力：[6EAH]←ファイルディスクリプタを示すテキストアドレス 出力：[152CH]←ファイル名 [152BH]←デバイス番号 [152AH]←デバイスタイプ} (00H：DISK　02H：COM 01H：CMT　03H：LPT)
14H	デバイスタイプの検出
15H	サブルーチンの呼出し　　入力：[A00H]←サブルーチンのエントリポインタ
16H	データアドレスの検出 入力：AL←変数名の頭文字/[6EA,6EB]変数名の先頭アドレス DL←変数名の型 DH←変数名の長さ－1 出力：[154EH]←変数格納域アドレス(オフセットアドレス) [1550H]←変数格納域アドレス(ベースアドレス)

(注) 表中の[　]で示したアドレスはセグメントアドレス＝0060Hを基準にしている
**　アドレスについては指定オフセット＋0，＋1の2バイトに格納する

コマンドコード	機　　能
17H 18H 19H 1AH	変数の内容を FAC に代入* FAC の内容を，指定アドレスへストアする* 文字変数格納位置の検出 次のトークンがカンマ(,)の時，それをスキップする
1BH	指定行から実行開始 入力：BX←指定行の先頭アドレス
1CH	指定行の次の行から実行開始 入力：[6ECH]←指定行の先頭アドレス
1DH	数値定数の表現の変換，VAL関数（文字表現→数値表現） 入力：[6EAH]←文字列の先頭アドレス 出力：FAC←数値*
1EH 1FH 20H 21H	ダイレクトモードエントリ テキストエディットのためのステータスをリセット PRINT USING 用編集 値の編集（STR＄関数）
22H	スペースをスキップ 入力：[6EAH]←テキストアドレス
23H	文の終端の検出 入力：SI←テキストアドレス
24H	バイナリ数値を文字列に変換 入力：FAC←数値データ BX←バッファポインタ
25H	カレントデバイスへの出力 入力：[1840H]←出力先コード(03H：プリンタ, 04H：CRT, その他：ファイル) [1842H]←文字のバッファのアドレス CX←文字数
26H 27H 28H 29H 2AH 2BH 2CH 2DH 2EH	Syntax error を表示 Type mismatch を表示 Illegal function call を表示 実数化　FAC←CSNG(FAC)* 倍精度化　FAC←CDBL(FAC)* 実数加算　FAC←FAC1＋FAC* 実数減算　FAC←FAC1－FAC* 実数乗算　FAC←FAC1×FAC* 実数除算　FAC←FAC1÷FAC*

コマンドコード	機　　能
2FH	実数比較 FAC← { −1：FAC1＜FAC* 0：FAC1＝FAC 1：FAC1＞FAC
30H	倍精度減算 FAC←FAC1−FAC*
31H	倍精度比較 FAC← { −1：FAC1＜FAC 0：FAC1＝FAC* 1：FAC1＞FAC
32H 33H	バイナリ数値を 8 進表現に変換 OCT$(FAC)* バイナリ数値を16進表現に変換 HEX$(FAC)*
34H	符号なし整数値を10進表現に変換 入力：AX←バイナリ数 BX←バッファのポインタ
35H	テキストアドレスを行番号に書きかえる
36H	テキストから1項目を抽出 入力：[6EAH]←テキストアドレス
37H 38H 39H	指定行のテキストアドレス検出 指定行のテキストアドレス検出 テキストのサーチ　入力：AX←行番号　出力：BX←テキストアドレス
3AH	数式の存在チェック 入力：[6EAH]←テキストアドレス 出力：キャリーフラグ←(0：数式あり　1：なし)
3BH	CRT への表示 入力：CX←文字数（含リモートBASICプロトコル）
3CH	CRTへの表示（CRTのみ）
3DH	1文字を CRT 表示 入力：AL←文字のコード
3EH 3FH 40H 41H 42H 43H	カーソルを行の左端へ移動 CR, LF を CRT に出力（含リモートBASICプロトコル） CR, LFを出力（CRTのみ） 3EH と同じ 符号なし整数化 FAC←INT(FAC)* 添字評価

コマンドコード	機　　　　能
44H 45H	ガベージコレクション 符号なし整数の実数化 FAC←CSNG(FAC)*
46H	"in △△△△（行番号）"を表示 　　入力：AX←行番号
47H	文のスキップ 　　入力：[6EAH]←テキストアドレス
48H 49H	DATA 文のスキップ 文字列定数のスキップ
4AH	数式の評価 　　入力：[6EAH]←数式の先頭アドレス 　　出力：FAC←結果*
4BH 4CH	キー，タイマからの割り込みをセンス COM, PEN からの割り込みをセンス
4DH	アスキー表現の1行を内部表現に変換 　　入力：CX←文字のバイト数，[1406H]←文字例の先頭アドレス
4EH	メモリスイッチ 　　入力：BX←SW番号(E2,E6,EA,EE,F2,F6,FA,FE) 　　出力：AL←スイッチの内容
4FH	未実装 RAM へのアクセスチェック 　　入力：AX←オフセットアドレス 　　　　　[750H]←セグメントアドレス 　　出力：RAM 未実装の時，Illegal function call を表示
50H	ストリングエリアの確保 　　入力：CX←文字列長(≦255)
51H	キーワードのサーチ 　　入力：[6EAH]←テキストアドレス 　　　　　AL←キーワード 　　出力：[6EAH]←キーワードのあるテキストアドレス
52H 53H 54H	キーボードから1行入力 　　ワールド座標からスクリーン座標へ変換(X 座標) 　　ワールド座標からスクリーン座標へ変換(Y 座標)
55H	シンボルテーブルのスキャン 　　入力：AL←スキャンする変数名の頭文字 　　出力：SI←変数名のポインタ
56H	配列テーブルのスキャン 　　入力：AL←スキャンする配列変数名の頭文字 　　出力：SI←配列変数名のポインタ

FAC(Floating Point Accumulator), FAC1, FACTYP

BASICインタープリタのワークエリア上に設けられているメモリ領域で，FAC, FAC 1のサイズはどちらもそれぞれ先頭アドレスは，次のようになります．

FAC　　：[1416H]
FAC 1　：[1420H]
FACTYP：[1414H]

FACには演算項に相当する数値データを，FAC 1には被演算項に相当する数値データを格納します．

FACTYPにはFAC, FAC 1に格納する数値データのタイプを示すコードを設定します．

型コード	型　名　称	記　　号
02H	整数型	INT
03H	文字列型	STR
04H	単精度実数型	SNG
05H	漢字文字列型	KANJI
06H	倍精度実数型	DBL

FAC, FAC 1へのデータ格納状態は，次の様になっています．

(先頭)

	B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8	バイト番号
(1)	仮　数							指数	倍精度実数型 (DBL)
(2)	未　使　用				仮　数			指数	単精度実数型 (SNG)
(3)	未　使　用				数　　値				整数型 (INT)
(4)	未　使　用				オフセットアドレス		セグメントアドレス		文字列型 (STR)
(5)	未　使　用				オフセットアドレス		セグメントアドレス		漢字文字列 (KANJI)

5 BIOS活用の手続き

ソフトウェア割り込みを用いて活用できるルーチンとして，BIOS*があります．BIOSには，**図3-1**に示したような種類があります．

ここでは，BIOSを活用するための手続きについて説明しますが，一例として，DISK BIOSを取り上げます．

DISK BIOSのベクタコードは，**図3-1**に示したように1BHです．DISK BIOSは，複数個のDISK BIOSコマンドとして系統化されており，個々のコマンドには**表5-8**に示すように，DISK BIOSコマンドコードが割り当てられています．

例えば，DISK BIOSの中で，コマンドコードmに対応するDISK BIOSコマンドをコールする場合の手続きは，以下に示す通りです．

①レジスタAHにコマンドコードを設定する

AH←m

②必要ならば所定のレジスタおよびメモリ領域にパラメータ値を設定する

③ソフトウェア割り込みの実行

INT　1BH

AH←m

なお，他のBIOSについても同様です．

表3-3 各種BIOSのベクタコード

ベクタコード	BIOS名称	備　　考
18H	キーボード/CRT BIOS	第1章6.3，第4章6参照
19H	RS-232C BIOS	第6章2.3参照
1AH	プリンタBIOS	第6章5.3参照
1BH	DISK BIOS	第5章3.1参照
1CH	カレンダー・タイマ BIOS	第1章7.3，第1章8.3参照
D1H	GP-IB BIOS	第6章3.2参照
33H	マウス BIOS	第6章4.2参照

* BIOS＝Basic Input Output System
周辺ハードウェアに対するデータの入出力を制御するための基本ソフトウェア

第4章

グラフィックス

CONTENT

1 概要

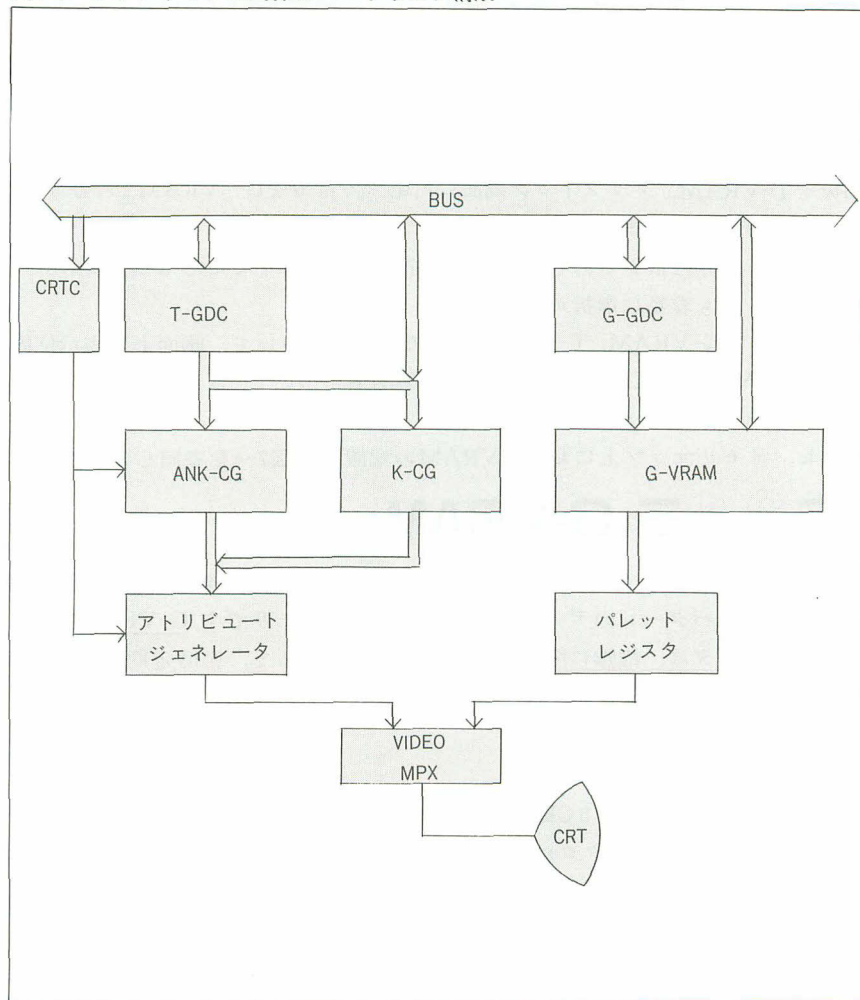
本章では, PC-98の持つ画面表示に関する機能, つまりグラフィックス機能について, ハードウェア, ソフトウェアの両面から解説していきます.

PC-98の画面表示モードには, テキストモードとグラフィックモードの2種類があります. 画面表示の最小単位が, 前者ではキャラクタ単位であるのに対して, 後者ではドット単位になっています.

PC-98のグラフィックス機能を支えているハードウェアの構成の概要を図4-1に示します.

PC-98のグラフィックス機能をフルに引き出すために十分理解しておかなければならない要素は, VRAM (T-VRAM, G-VRAM), GDC (T-GDC, G-GDC), CRTC, CG(ANK-CG, K-CG)です. これらは, VRAMを除いて, すべてコントローラとしての機能を持つ専用LSIです. コントローラがVRAMに格納されているデータに基づいてCRTに画面表示を行っています. 本章の前半で, これらハードウェアについての解説を行います.

また, PC-98には, グラフィックス関係のハードウェアの持つ能力をより簡単な手続きで最大限に引き出すための基本ソフトウェアとして, CRT BIOS, グラフィックLIOが用意されています. 本章の後半では, これら基本ソフトウェアの活用方法について解説を行います.

図4-1　グラフィックス関係のハードウェア構成

2 VRAM

PC-98のCRT表示モードには，テキストモードとグラフィックモードの２種類があります．画面表示するためのデータ(文字コード，ビットパターンetc)を書き込む画面表示用メモリをVRAM*と呼び，特にテキスト画面に対応する部分をT-VRAM，グラフィック画面に対応する部分をG- VRAMと呼びます．それぞれの画面に専属のVRAMを備えたことにより，テキスト画面とグラフィック画面を独立に扱えるので，一方だけを単独で表示することも重ね合わせて表示することも容易に選択可能です．

以下では，G-VRAM，T-VRAMのメモリ構成について，画面上の表示位置と，これらVRAMのCPUアドレスとの対応関係について，さらに，VRAMに書き込むデータの形式について説明します．

なお，メモリマップ上におけるVRAMの配置は，**図2-4**を参照して下さい．

2.1 G-VRAM

G-VRAMは，グラフィック画面に表示させたいドットパターンに１対１に対応したビットパターンのデータを書き込むためのメモリです．G-VRAMに書き込まれたデータは，描画に関するハードウェアによって，常時定期的に読み出され，ドットパターンがグラフィック画面に表示されます．

(1)CRTの表示モード

グラフィック時におけるCRTの表示モードには，１画面を構成するドット数の違いと，カラーかモノクロかの違いによって，モノクロ200モード，カラー200モード，モノクロ400モード，カラー400モードの４種類あります．各モード名称の数値は，画面の縦方向のドット数を表しています．これは，カラーとモノクロの違いと，分解能の違いに着目した分類です．

グラフィックモードでは，さらに８色グラフィックモード（標準グラフィックモード）と16色グラフィックモード（拡張グラフィックモード）に分類できます．前者ではカラーパレットが８種類であるのに対して，後者では16種類で

---

*　VRAM＝ビデオRAM，T－VRAM＝テキストVRAM，G－VRAM＝グラフィックVRAM

す．それにともない，前者では色の自由度が８色であり，後者では4096色の中の任意の16色となります．以上述べた各モードを**表4-1**にまとめて示します．

表4-1　CRTの表示モードの種類（グラフィック時）

	モード名	画面のドット構成	1画面の所要メモリ(KB)	1ビットの所要ビット	備　考
8色モード	モノクロ200モード	640×200	16	1	
	カラー200モード	640×200	48（16×3）	3	8色
	モノクロ400モード	640×400	32（16×2）	1	
	カラー400モード	640×400	96（32×3）	3	8色
16色モード	モノクロ200モード	640×200	16	1	
	カラー200モード	640×200	64（16×4）	4	4096色中の16色
	モノクロ400モード	640×400	32（16×2）	1	
	カラー400モード	640×400	128（32×4）	4	4096色中の16色

＊　８色モード＝８色グラフィックモード（標準グラフィックモード）
16色モード＝16色グラフィックモード（拡張グラフィックモード）（E/F/Mではサポートされない）

⑵メモリマップ

G-VRAMのメモリマップを**図4-2**に示します．G-VRAMは，標準G-VRAMと拡張G-VRAMで構成されています．それぞれ割り当てられているCPUアドレスは，A8000H～BFFFFHとE0000H～E7FFFHで，メモリサイズは，96Kバイトと32Kバイトです．メモリマップで見るかぎりG-VRAMは128Kバイトですが，実際は同一アドレスに対してもう１組のG-VRAMを重ねて割り付けているので，２倍の256Kバイトになっています．この２組のG-VRAMをG-VRAM(1)，G-VRAM(2)と呼びますが，このように構成しても，スイッチの切り替えにより，２組のG-VRAMの一方だけを選択してアクセスするので混乱は生じません*．

＊　機種によりＧ－ＶＲＡＭの実装状態が異なる．

メモリ名 ＼ 機種		メモリサイズ（KB）			
		ＵＶ	VF／VM	Ｕ	Ｅ／Ｆ／Ｍ
G-VRAM(1)	標準G-VRAM	96	96	96	96
	拡張G-VRAM	32	(32)	(32)	—
G-VRAM(2)	標準G-VRAM	96	96	—	96
	拡張G-VRAM	32	32	—	—
合　計		256	192(64)	128	192

注）（　）内はオプションの16色グラフィックボード追加による増加分
ＵＶは16色グラフィックボード標準実装
ＵはＧ－ＶＲＡＭ(２)がない

第４章 グラフィックス

以下では，G-VRAM(1)を想定して説明しますが，G-VRAM(2)についてもまったく同様です．

図4-2に示すように，96Kバイトの標準G-VRAMを3等分し，それぞれBプレーン，Rプレーン，Gプレーンと呼ぶことにします．32Kバイトの拡張G-VRAMをIプレーンと呼ぶことにします．さらに，各プレーンを2等分割して，PB1プレーン，PB2プレーンのように呼ぶことにします．以下では，**表4-1**に示した各モードでG-VRAMがどのように使われるかを説明しますが，まず8色グラフィックモードと16色グラフィックモードに分けて行います．

なお，**図4-2**には，CPUアドレスの横にGDCアドレスを併記しています．前者がメモリ1バイトごとに割り当てられたバイト形式アドレスであるのに対して，後者はメモリ1ワード（2バイト）ごとに割り当てられたワード形式アドレスです．PC-98では，描画速度の向上を図るため，描画制御用LSIであるGDCを使用しています．このGDCがG-VRAMにアクセスする場合に参照するアドレスがGDCアドレスです．GDCについては，本章4で説明しています．

(i) 8色グラフィックモード（標準グラフィックモード）

8色グラフィックモードでは，標準G-VRAMのみを使用し，拡張G-VRAMは使用しません．カラー400モード，カラー200モード，モノクロ400モード，モノクロ200モードの場合に分けて，G-VRAMの使い方について説明します．

①カラー400モード

Bプレーン，Rプレーン，Gプレーンを使用します．640×400ドットの画面上の1ドットに対応するBプレーン，Rプレーン，Gプレーン上の1ビットをそれぞれb，r，gとします．ドットの色は，ビットパターンb r gで選択される番号のカラーパレットによって指定されます．カラーパレットの番号とカラーコードが一致しているときにはb，r，gが3原色（B，R，G）の輝点のスイッチの機能を果たすことになります．この意味でBプレーン，Rプレーン，Gプレーンという呼び方を使っています．カラー400モードの画面枚数は1で，その名称をPb＋Pr＋Pgで表します．

②カラー200モード

B1プレーン，R1プレーン，G1プレーンまたはB2プレーン，R2プレーン，G2プレーンを使用します．カラー200モードの画面枚数は2で，その名称をPb1＋Pr1＋Pg1，Pb2＋Pr2＋Pg2で表します．前者の場合を例にとり640×200ドットの画面上の1ドットに対応するB1プレーン，R1プレーン，G1プレーン上の1ドットをそれぞれb，r，gとします．以下の色の指定方法は，①カラー400モー

図4-2　G-VRAMのメモリマップおよび各モードにおける画面構成

G-VRAMのアドレス						表示モード名称				
GDCアドレス	CPUアドレス 偶数アドレス	CPUアドレス 奇数アドレス	メモリサイズ (KB)	プレーン名称		モノクロ200	モノクロ400	カラー200	カラー400	
4000H ∫	A8000H ∫	A8001H ∫	16／16	Bプレーン／$\overline{B}$プレーン	B1プレーン／$\overline{B1}$プレーン	PB1／$\overline{PB1}$	PB／$\overline{PB}$	Pb1／$\overline{Pb1}$	Pb／$\overline{Pb}$	画面名称
5F40H ∫	ABE80H ∫	ABE81H ∫	16／16		B2プレーン／$\overline{B2}$プレーン	PB2／$\overline{PB2}$		Pb2／$\overline{Pb2}$		
8000H ∫	B0000H ∫	B0001H ∫	16／16	Rプレーン／$\overline{R}$プレーン	R1プレーン／$\overline{R1}$プレーン	PR1／$\overline{PR1}$	PR／$\overline{PR}$	Pr1／$\overline{Pr1}$	Pr／$\overline{Pr}$	
9F40H ∫	B3E80H ∫	B3E81H ∫	16／16		R2プレーン／$\overline{R2}$プレーン	PR2／$\overline{PR2}$		Pr2／$\overline{Pr2}$		
C000H ∫	B8000H ∫	B8001H ∫	16／16	Gプレーン／$\overline{G}$プレーン	G2プレーン／$\overline{G1}$プレーン	PG1／$\overline{PG1}$	PG／$\overline{PG}$	Pg1／$\overline{Pg1}$	Pg／$\overline{Pg}$	
DF40H ∫	BBE80H ∫	BBE81H ∫	16／16		G2プレーン／$\overline{G2}$プレーン	PG2／$\overline{PG2}$		Pg2／$\overline{Pg2}$		
0000H ∫	E0000H ∫	B0001H ∫	16／16	Iプレーン／$\overline{I}$プレーン	I1プレーン／$\overline{I1}$プレーン	PI1／$\overline{PI1}$	PI／$\overline{PI}$	Pi1／$\overline{Pi1}$	Pi／$\overline{Pi}$	
1F40H ∫	E3E80H ∫	E3E81H ∫	16／16		I2プレーン／$\overline{I2}$プレーン	PI2／$\overline{PI2}$		Pi2／$\overline{Pi2}$		
8色グラフィックモード→			96／96			6／6	3／3	2／2	1／1	画面枚数
16色グラフィックモード→			128／128			8／8	4／4	2／2	1／1	

注）図中の"／"の左側，右側の記述はそれぞれG-VRAM(1)，G-VRAM(2)に対応する．

＊　モノクロモードでは，各々が1画面の名称を表す．
カラーモードでは，3プレーン(8色グラフィックモード時)または，4プレーン(16色グラフィックモード時)の合成で1画面の名称を表す．
（例）Pb＋Pr＋Pg＋Pi（16色グラフィックモードでカラー400モード時）

ドと同様なので省略します.

③モノクロ400モード

1つのプレーン（B，R，Gのいずれか）で1画面を表現します．モノクロ400モードの画面枚数は3で，その名称はPB，PR，PGで表します．

④モノクロ200モード

1つのプレーン（B1，R1，G1，B2，R2，G2のいずれか）で1画面を表現します．モノクロ200モードの画面枚数は6で，その名称をPB1，PR1，PG1，PB2，PR2，PG2で表します．

(ii) 16色グラフィックモード（拡張グラフィックモード）

拡張グラフィックモードでは，標準G-VRAMに加えて，拡張G-VRAMも使用可能です．カラー400モード，カラー200モード，モノクロ400モード，モノクロ200モードの場合に分けて，G-VRAMの使い方について説明します．

①カラー400モード

Bプレーン，Rプレーン，Gプレーン，Iプレーンを使用します．640×400ドットの画面上の1ドットに対応するBプレーン，Rプレーン，Gプレーン，Iプレーンをそれぞれb，r，g，iとします．

ドットの色はビットパターン b　r　g　i で選択される番号のカラーパレットによって指定されます．カラー400モードの画面枚数は1で，その名称はPb＋Pr＋Pg＋Piで表します．

②カラー200モード

B1，R1，G1，I1プレーンまたはB2，R2，G2，I2プレーンを使用します．カラー200モードの画面構成は2で，その名称をPb1＋Pr1＋Pg1＋Pi1，Pb2＋Pr2＋Pg2＋Pi2で表します．前者の場合を例にとり，640×200ドットの画面上の1ドットに対応するB1，P1，G1プレーン上の1ビットをそれぞれb，r，g，iとします．以下の指定方法は，①カラー400モードと同様なので省略します．

③モノクロ400モード

1つのプレーン（B，R，G，Iのいずれか）で1画面を表現します．モノクロ400モードの画面構成は4で，その名称をPB，PR，PG，PIで表します．

④モノクロ200モード

1つのプレーン（B1，R1，G1，I1，B2，R2，G2，I2のいずれか）で1画面を表現します．モノクロ200モードの画面枚数は8で，その名称をPB1，PR1，PG1，PI1，PB2，PR2，PG2，PI2で表します．

［G-VRAM(1)とG-VRAM(2)の切り替え］

G-VRAM(1)とG-VRAM(2)の選択のためのスイッチの機能を果たすものとして，2種類のI/Oポートアドレスが割り当てられています．データをG-VRAMに書き込む場合の選択と，データをG-VRAMから読み出す場合の選択では，使用するI/Oポートアドレスが異なり，それぞれA6H，A4Hです．いずれの場合も，I/Oポートアドレスに数値00Hを出力すれば，G-VRAM(1)が選択され，01Hを出力すれば，G-VRAM(2)が選択されます．以上，述べた内容をまとめると，G-VRAM(1)，(2)を選択・指定するためのプログラムは，以下のように表現できます．

モード ＼ メモリ名	G-VRAM(1)	G-VRAM(2)
描画 A6　入力モード (G-VRAMへ書き込み)	MOV AL, 00H OUT A6H, AL	MOV AL, 01H OUT A6H, AL
表示 A4　出力モード (G-VRAMから読み出し)	MOV AL, 00H OUT A4H, AL	MOV AL, 01H OUT A4H, AL

(3)グラフィック画面とG-VRAMの対応関係

グラフィック画面上のドットパターンと，G-VRAM上のデータのビットパターンの対応関係を**図4-3**に示します．ただし，モノクロ200モードの場合における画面PB1を例としています．**図4-3(1)**の図②は，グラフィック画面を640×200ドットに分割した状態を表したもので，画面のドットパターンとデータのビットパターンの対応関係を示しています．**図4-3(1)**の図①はデータが格納されているG-VRAMのCPUアドレスを示しています．

ここで，注意すべき点を2点だけ列記しておきます．

① 8ビットデータのビット番号の増加方向と，グラフィック画面上のドットのX座標値の増加方向とが，ちょうど逆になっている．

② 汎用メモリとG-VRAMの間でデータ転送をする場合，CPUで直接アクセスする方法と，GDCを介してアクセスする方法の2種類がある．後者の場合には，汎用メモリ上のデータのビットパターンとG-VRAM上のデータのビットパターンが**図4-3(2)**に示すように異なるので，汎用メモリからG-VRAMへ転送する際には注意を要する．

図4-3(1) グラフィック画面の表示位置とG-VRAMのCPUアドレスの対応関係

① G-VRAMのCPUアドレス

Y				
Y=0	A8000H	A8001H		A804FH
1	A8050H	A8051H		A809FH
≈	≈	≈	≈	≈
199	ABE30H	ABE31H		ABE7FH

② G-VRAMのデータのビットパターンと画面上のドットパターンの対応関係

X=0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 … 639

ドット座標	X=0〜7	X=8〜15		〜639
Y=0	MSB b6 b5 b4 b3 b2 b1 LSB	MSB … LSB		MSB … LSB
1	MSB … LSB	MSB … LSB		MSB … LSB
≈	≈	≈	≈	≈
199	MSB … LSB	MSB … LSB		MSB … LSB

注）図②はモノクロ200モードの画面ＰＢ１のドットパターンを表していて，Ｘ，Ｙが画面上のドット座標である．

　図①は，Ｇ－ＶＲＡＭのどのアドレスのデータが図②に示した図面上のドットパターンに対応するかを示している．

図4-3(2) データ転送の方式の相違に伴う汎用メモリ上のデータのビットパターンとG-VRAM上のデータのビットパターンの対応関係

＊（注）ＧＤＣを介した場合には，汎用メモリ上のデータのビットパターンとG-ＶＲＡＭ上のデータのビットパターンとでは，ビットの上位，下位が反転している。

2.2 T-VRAM

T-VRAMは,テキスト画面に表示させたい文字に対応するデータを書き込むためのメモリであり,文字コード領域とアトリビュート領域に分類できます.前者には,その名前のように,表示させたいANK文字*のASCIIコードや漢字JISコードなどを書き込み,後者には,文字を表示する場合の属性(文字の色,リバース表示,ブリンク表示,etc)を規定するデータ,つまりアトリビュートデータを書き込みます.

T-VRAMに格納されているデータは,描画に関するハードウェアによって常時定期的に読み出され,指定の属性で指定の文字がテキスト画面に表示されます.

(1)テキスト画面の表示モード

テキスト画面の表示モードには4種類あり,それぞれ1画面当りの行数と1行当りの桁数との組み合わせが異なっていて,それを**表4-2**にまとめて示します.

なお,ここでは,80×25モードと80×20モードを総称して80字モードと呼びます.同様に,40字モードあるいは20字モード,25字モードという呼び方も使用します**.

表4-2 テキスト画面の表示モード (注)

モード名	行数/画面	字数/行	1桁(1文字)分の表示領域のドット構成	備考
80×25モード	25行	80字	8×16ドット	
80×20モード	20行	80字	8×20ドット	
40×25モード	25行	40字	16×16ドット	
40×20モード	20行	40字	16×20ドット	

(注) 画面の表示モードが,640×400ドットである場合を想定している.
表4-1参照.この1文字分の表示領域のことを,ボディフェースと呼ぶ.

* ANK文字=英数カタカナ文字(アルファベット,ニューメリック,カタカナ)
** 以下に各モードを示します。

80字モード	80×20, 80×25 モードの総称
40字モード	40×20, 40×25 モードの総称
20行モード	80×20, 40×20 モードの総称
25行モード	80×25, 40×25 モードの総称

第4章 グラフィックス

(2)メモリマップ

T-VRAMのメモリマップを図4-4に示します．

T-VRAMに割り当てているCPUアドレスは，A0000HからA3FFFHであり，このアドレス領域に対応するメモリサイズは12Kバイトです．このサイズはテキスト画面2画面分に相当します．BASICレベルでは2画面の中の1画面しか使用していませんが，マシン語レベルでは2画面とも活用することができます．

ここで，T-VRAMのメモリ構成に関して留意すべき点を2点列記しておきます．

① 1ワード（2バイト）を基本単位としてメモリを使用している．

② アトリビュート領域では，偶数アドレスに割り当てられている下位バイトのみ使用する．奇数アドレスには，メモリが実装されていない．

2バイトを基本単位としている理由は，2バイトデータであるJISコードで表現される日本字*を，1バイトデータであるASCIIコードで表現されるANK文字とほぼ同等の扱いでテキスト画面に表示させることを考慮したためであると考

図4-4　T-VRAMのメモリマップ

T-VRAMのアドレス			T-VRAMの使用状況	
GDCアドレス	CPUアドレス			
	偶数アドレス	奇数アドレス	偶数アドレス	奇数アドレス
0000H 〜 07FFH	A0000H 〜 A0FFEH	A0001H 〜 A0FFFH	第1画面の文字コード領域（約4KB）	
0800H 〜 0FFFH	A1000H 〜 A1FFEH	A1001H 〜 A1FFFH	第2画面の文字コード領域（約4KB）	
1000H 〜 17FFH	A2000H 〜 A2FFEH	A2001H 〜 A2FFFH	第1画面のアトリビュート領域（約2KB）	メモリ未実装
1800H 〜 1FFFH	A3000H 〜 A3FFEH	A3001H 〜 A3FFFH	第2画面のアトリビュート領域（約2KB）	メモリ未実装

（注）各領域とも，厳密に言えば末尾に未使用部分があるが，
ここでは，未使用部分も区別しないで各領域に含めて表現している．

＊ 漢字やひらがな等，2バイトのJISコードで表現される文字を日本字と呼ぶ．

えられます．

なお，**図4-4**には，CPUアドレスの横にGDCアドレスを併記しています．描画制御用LSIであるGDCがT-VRAMにアクセスする場合に参照するアドレスがGDCアドレスです．GDCアドレスについては，本章の3節で説明しています．

⑶テキスト画面とT-VRAMの対応関係

テキスト画面上の文字表示位置と，T-VRAMのCPUアドレスとの対応関係を**図4-5**に示します．**図4-5**①は，80字モードの場合に，テキスト画面がボディフェースを単位として分割されている様子を示しています．各々のボディフェースに割り当てられている数字をテキストアドレスと呼ぶことにします．80字モードの場合には，1つのテキストアドレスに対してアトリビュート領域のメモリ1バイトと文字コード領域のメモリ2バイトが対応します．その様子を**図4-5**③，④に示しています．

なお，**図4-5**②は，40字モードの場合におけるテキスト画面の分割の様子を示しています．40字モードにおけるボディフェースの横方向のドット数は，80字モードのそれに比べ2倍になっています．**図4-5**②，**図4-5**④を対照すれば明らかなように，40字モードでは1つのテキストアドレスに対して，文字コード領域のメモリ4バイトが対応します．

アトリビュート領域，文字コード領域に格納するデータの形式については，次項⑷，⑸で説明します．

図4-5　テキスト画面上のテキストアドレスとT-VRAMのCPUアドレスとの対応関係

①テキスト画面上のテキストアドレス*（80字モード）

0	0	1	2	3	≈	78	79
1	80	81	82	83	≈	158	159
≈							
25または20							

②テキスト画面上のテキストアドレス（40字モード）

0	0	1	≈	39
1	40	41	≈	79
≈		~		
25または20				

③アトリビュート領域に対するCPUアドレス

0	A2000H	A2002H	A2004H	A2006H	≈	A209CH	A209EH
1							
≈							
25または20							

④文字コード領域に対するCPUアドレス

0	A0000H	A0001H	A0002H	A0003H	A0004H	A0005H	A0006H	A0007H	≈	A009CH	A009DH	A009EH	A009FH
1													
≈													
25または20													

⑤文字コード領域に対するGDCアドレス

0	0000H	0001H	0002H	0003H	≈	004EH	004FH
1	0050H						
≈							
25または20							

＊1つのテキストアドレスに対応する表示領域をボディフェースと呼ぶ．図4-13(2)参照．

⑷アトリビュート領域のデータ形式

T-VRAMのアトリビュート領域に格納するアトリビュートデータの形式について説明します．

アトリビュート領域では，偶数アドレスのメモリだけを使用しています．テキスト画面に文字を表示するときの画面属性をアトリビュートデータが規定しています．アトリビュートデータの各ビットが規定する属性内容を**表4-3**に示します．

BASICレベルでは，カラーモードのときにリバース表示やブリンク表示を指定するコマンドが準備されていませんが，アトリビュート領域にデータを直接格納することによって，カラーモードでのリバース表示，ブリンク表示など，多彩な表現を楽しむことができます．

表4-3 アトリビュートデータで指定できる画面属性

ビット番号 / ビットの値	b_7 (注1)	b_6	b_5	b_4 (注2)	b_3	b_2	b_1	b_0
1	G	R	B	垂線表示又は簡易グラフ	アンダーライン表示	リバース表示	ブリンク表示	ノーマル
0				ノーマル	ノーマル	ノーマル	ノーマル	シークレット

（注1）a. カラーモニタの時には，$b_7b_6b_5$の3ビットは表示する文字の色を指定する．3ビットのパターンと，それに対応する色を以下に示す．

b_7 （G）	0	0	0	0	1	1	1	1
b_6 （R）	0	0	1	1	0	0	1	1
b_5 （B）	0	1	0	1	0	1	0	1
色　彩	黒	青	赤	紫	緑	水色	黄色	白

b. モノクロモニタの時には，$b_7b_6b_5$の3ビットの値が，表示する文字の濃淡を8階調で指定する．111の時，最も明るくなる．

（注2）ビットb_4＝1とした時，垂線表示と簡易グラフのいずれが選択されるかは，他のスイッチの状態に依存する．具体的には，GDCモードレジスタのビットb_0がそのスイッチに相当する．

GDCモードレジスタのビットb_0＝0 → 垂線表示
　　　　　　　　　　　　　　1 → 簡易グラフ

GDCモードレジスタについては，表4-7参照．

(5)文字コード領域のデータ形式

T-VRAMの文字コード領域に格納するデータ形式について説明します.

以下では，テキスト画面が80字モードに設定されているものとして述べていきます.まず，1バイトのASCIIコードで表現されるANK文字の場合について，続いて，2バイトのJISコードで表現される日本字の場合について説明します.

①ANK文字を表示する場合

テキスト画面にANK文字を表示する際に，T-VRAMの文字コード領域に格納すべきデータの形式について説明します. 80字モードのとき，ANK文字は，**図4-5①**に示したテキストアドレス1つ分に相当する領域に表示されます.このとき，ANK文字のASCIIコードは，**図4-5④**に示した文字コード領域の中で，上記テキストアドレスに対応する2バイト分のメモリに格納されます. 具体例として，**図4-5①**におけるテキストアドレス0の領域にANK文字を表示する場合を考えて，このときのデータの格納状態を**図4-6**に示します.

なお，40字モードのときは，ANK文字は，**図4-5②**で示したテキストアドレスの1つ分に相当する領域に表示されます. このとき，ANK文字のASCIIコードは，**図4-5④**に示した文字コード領域に対応する4バイト分のメモリに格納されます. 具体例として，**図4-5②**におけるテキストアドレス0の領域にANK文字を表示させる場合を考えて，このときのデータの格納状態を**図4-7**に示します.

図4-6　文字コード領域のデータ格納状態（80字モード時にANK文字をテキストアドレス0の位置に表示させる場合）

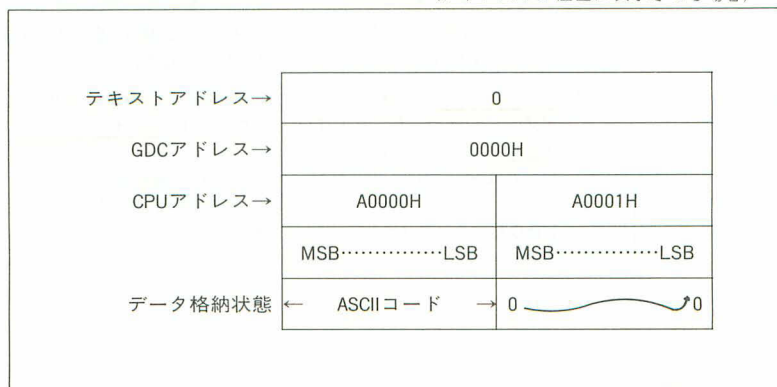
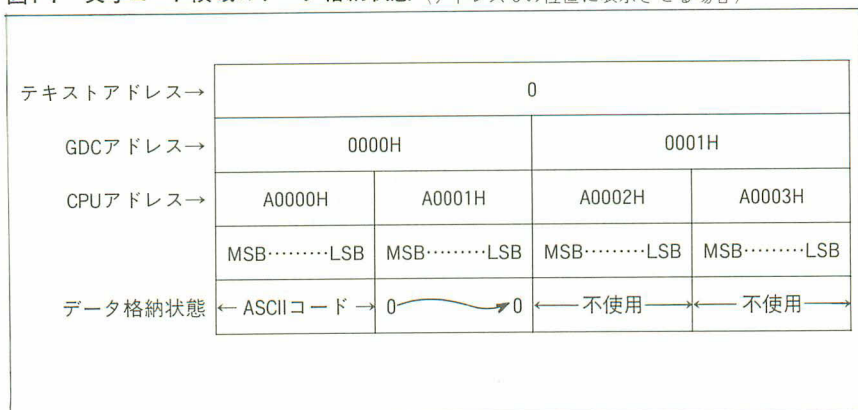



図4-7　文字コード領域のデータ格納状態（40字モード時にANK文字をテキストアドレス0の位置に表示させる場合）

テキストアドレス→	0			
GDCアドレス→	0000H		0001H	
CPUアドレス→	A0000H	A0001H	A0002H	A0003H
	MSB………LSB	MSB………LSB	MSB………LSB	MSB………LSB
データ格納状態	← ASCIIコード →	0→0	←不使用→	←不使用→

②日本字を表示する場合

テキスト画面に日本字を表示する際にT-VRAMの文字コード領域に格納すべきデータの形式について説明します．80字モードのとき，日本字は**図4-5①**に示したテキストアドレスの２つ分に相当する領域に表示されます．このとき，日本字のJISコードは**図4-5④**に示した文字コード領域の中で，上記２つのテキストアドレスに対応する４バイト分のメモリに格納されます．具体例として，**図4-5①**におけるテキストアドレス０，１の領域に「技」という日本字（漢字）を表示させる場合を考えて，このときのデータの格納状態を**図4-8**に示します．

なお，40桁モードのときに，日本字は**図4-5②**で示したテキストアドレスの２つ分に相当する領域に表示されます．このとき，日本字のJISコードは，**図4-5④**に示した文字コード領域の中で，上記２つのテキストアドレスに対応する８バイト分のメモリに格納されます．具体例として，**図4-5②**におけるテキストアドレス０，１の領域に「技」という日本字を表示させる場合を考えて，このときのデータの格納状態を**図4-9**に示します．

第4章　グラフィックス

図4-8　文字コード領域のデータ格納状態 (80字モード時に日本字「技」をテキストアドレス0,1の位置に表示させる場合)

テキストアドレス→	0		1	
GDCアドレス→	0000H		0001H	
CPUアドレス→	A0000H	A0001H	A0002H	A0003H
	MSB………LSB	MSB………LSB	MSB………LSB	MSB………LSB
データ格納状態	←DATA1→	←DATA2→	←DATA1→	←DATA3→

（注）漢字「技」のJISコード＝35 3BH（35→JIS上位バイト，3B→JIS下位バイト）

図中のDATA1・2・3の定義を以下に示す。

DATA1＝JIS上位バイト－20H＝35H－20H＝15H
DATA2＝JIS下位バイト　　＝3BH
DATA3＝JIS下位バイト＋80H＝3BH＋80H＝BBH

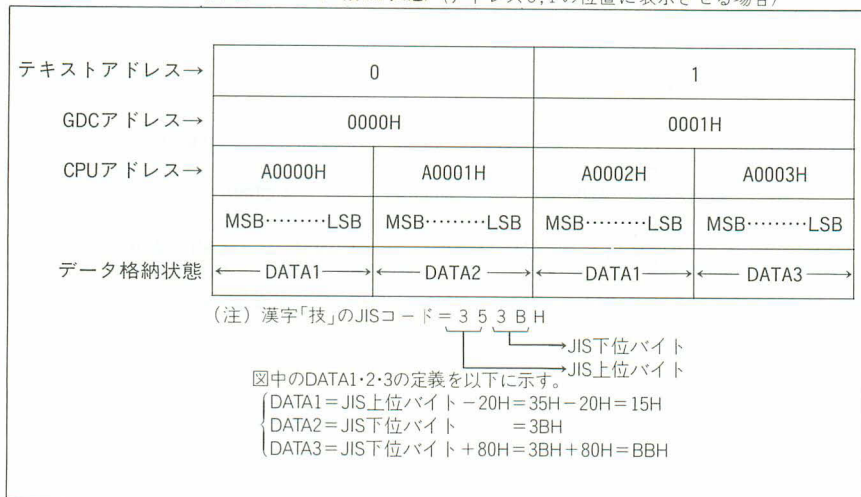
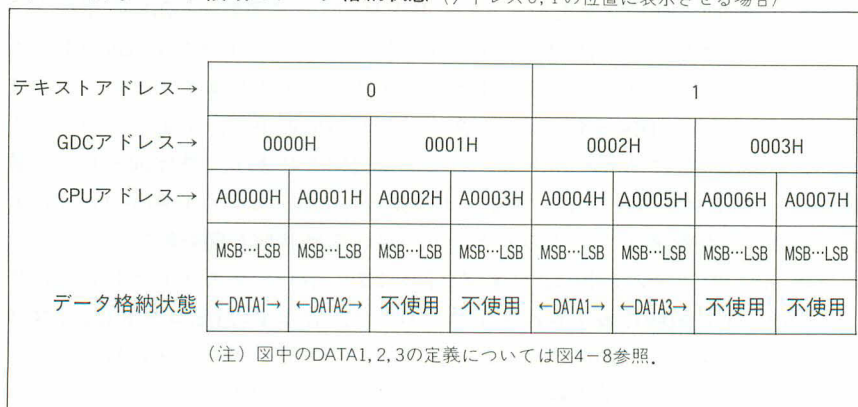



図4-9　文字コード領域のデータ格納状態 (40桁モード時に日本字「技」をテキストアドレス0,1の位置に表示させる場合)

テキストアドレス→	0				1			
GDCアドレス→	0000H		0001H		0002H		0003H	
CPUアドレス→	A0000H	A0001H	A0002H	A0003H	A0004H	A0005H	A0006H	A0007H
	MSB…LSB	MSB…LSB	MSB…LSB	MSB…LSB	MSB…LSB	MSB…LSB	MSB…LSB	MSB…LSB
データ格納状態	←DATA1→	←DATA2→	不使用	不使用	←DATA1→	←DATA3→	不使用	不使用

（注）図中のDATA1, 2, 3の定義については図4－8参照.

3 GDC

GDC*とは，PC-98の高速かつ多彩なCRT表示機能を実現するうえで中枢的働きをなしているLSI（μPD7220A）です．

GDCの主な機能を数例挙げてみると，

①画面表示をさせたいデータをVRAMへ書き込む．あるいは逆にVRAMのデータを読み出す．

②画面の上下スクロールおよび左右スクロールを行う．

③拡大表示を行う．

④円，直線，矩形の高速描画を行う．

⑤カーソル表示に関する制御を行う．

⑥ライトペンの制御を行う．

などがあります．これ以外にも機能を豊富に備えていますが，いずれの機能もGDCに対してI／O制御命令を与えることによって，選択・設定できます．具体的には，GDCの制御用に割り当てられているI／Oポートアドレスを介して制御データを入出力することで命令を与えます．I／O制御命令についての詳細は，3.1で述べます．

GDCは，CPUを介することなく，T-VRAMやG-VRAMに直接アクセスして描画作業の制御を行うことができるので，GDCの導入によって描画時におけるCPUの負荷が低減されるとともに，描画速度も著しく向上しています．VRAMにはGDCが直接アクセスできるように，CPUアドレスに加え“GDCアドレス”を割り当てています．前者がメモリ1バイトごとに割り当てられたバイト形式アドレスであるのに対して，後者はメモリ1ワード（2バイト）ごとに割り当てられたワード形式アドレスです．

PC-98は，このGDCを2個搭載しており，テキスト画面用，グラフィック画面用に機能分担させて使用しています．2個のGDCをそれぞれT-GDC（テキスト画面用GDC），G-GDC（グラフィック画面用GDC）と呼ぶことにします．2つのGDCは，LSI自体としてはまったく同一ですが，それぞれの分担を果たすのに必要な機能だけを実現する形で実装されているので，両者の機能は同一では

＊GDC＝Graphic Display Controler

ありません.

以下では，２つのGDCを機能させるためのＩ／Ｏ制御命令について説明し，次にＩ／Ｏ制御命令を組み合わせて作成したサンプルプログラムを示して解説します.

なお，画面モードとGDCの関係を表4-4に示します.

表4-4　画面モードとGDCの関係

<table>
<tr><th colspan="3">表示状態</th><th colspan="6">設定値</th></tr>
<tr><th>CRT</th><th>グラフィックモード</th><th>表示プレーン</th><th>GDC L／F</th><th>GDC L／R</th><th>GDC SAD</th><th>パレットレジスタ</th><th>Mode F／F ビット1</th><th>Mode F／F ビット4</th></tr>
<tr><td rowspan="6">高解像CRT</td><td rowspan="2">カラー640×200</td><td>Pb1+Pr1+Pg1</td><td rowspan="6">400</td><td rowspan="2">2</td><td>0</td><td rowspan="3">各コードのRGB</td><td rowspan="3">0</td><td rowspan="2">1</td></tr>
<tr><td>Pb2+Pr2+Pg2</td><td>1F40H</td></tr>
<tr><td>カラー640×400</td><td>Pb+Pr+Pg</td><td>1</td><td>0</td><td>0</td></tr>
<tr><td rowspan="2">モノクロ640×200</td><td>Pb1／$\overline{Pb1}$
Pr1／$\overline{Pr1}$
Pg1／$\overline{Pg1}$
(Pi1／$\overline{Pi1}$)</td><td rowspan="2">2</td><td>0</td><td rowspan="3">画面合成コード</td><td rowspan="3">1</td><td rowspan="2">1</td></tr>
<tr><td>Pb2／$\overline{Pb2}$
Pr2／$\overline{Pr2}$
Pg2／$\overline{Pg2}$
(Pi2／$\overline{Pi1}$)</td><td>1F40H</td></tr>
<tr><td>モノクロ640×400</td><td>Pb1／$\overline{Pb1}$
Pr1／$\overline{Pr1}$
Pg1／$\overline{Pg1}$
(Pi1／$\overline{Pi1}$)</td><td>1</td><td>0</td><td>0</td></tr>
<tr><td rowspan="4">標準CRT</td><td rowspan="2">カラー640×200</td><td>Pb1+Pr1+Pg1</td><td rowspan="4">200</td><td rowspan="4">1</td><td>0</td><td rowspan="2">各コードのRGB</td><td rowspan="2">0</td><td rowspan="4">0</td></tr>
<tr><td>Pb2+Pr2+Pg2</td><td>1F40H</td></tr>
<tr><td rowspan="2">モノクロ640×400</td><td>Pb1／$\overline{Pb1}$
Pr1／$\overline{Pr1}$
Pg1／$\overline{Pg1}$
(Pi1／$\overline{Pi1}$)</td><td>0</td><td rowspan="2">画面合成コード</td><td rowspan="2">1</td></tr>
<tr><td>Pb2／$\overline{Pb2}$
Pr2／$\overline{Pr2}$
Pg2／$\overline{Pg2}$
(Pi2／$\overline{Pi2}$)</td><td>1F40H</td></tr>
</table>

（）内は，16色グラフィックボード接続時に有効.
ＧＤＣの設定については表4-9，Ｍｏｄｅ　Ｆ/Ｆは表4-7をそれぞれ参照.
画面合成については，図4-10を参照.

≡3.1≡ T-GDCのI/O制御命令

T-GDCの機能を選択・設定するためのＩ／Ｏ制御命令について説明します．T-GDCの制御用に割り当てられているＩ／Ｏポートの種類は６種類であり，そのアドレスは60Ｈ，62Ｈ，64Ｈ，68Ｈ，6AＨ，6CＨです．このＩ／Ｏポートを介して制御データを出力することによって，T-GDCの制御を行っています．T-GDCのＩ／Ｏ制御命令をまとめて表4-5に示します．表には，各Ｉ／Ｏ制御命令の機能，使用するＩ／Ｏポートアドレス，および制御データの形式を示しています．

表4-4に示したＩ／Ｏ制御命令の各々について，より詳しく解説していきます．

①ライトコマンド命令，ライトパラメータ命令，リードデータ命令

ライトコマンド命令でGDCコマンドをＩ／Ｏポートアドレス62Ｈに出力することによって，GDCの多彩な機能の中から目的とする機能を選択・設定できます．多数あるGDCコマンド*の中には，GDCコマンドコードを出力しただけでは完結しないコマンドがあり，何種類かのGDCパラメータも与えなければなりません．この場合には，ライトパラメータ命令を併用することになります．

また，GDCコマンドの中には，GDCコマンドコードを出力した後に引き続い

表4-5 T-GDCのI/O制御命令

I/O制御命令	I/Oポートアドレス	I/O	制御データ $b_7\ b_6\ b_5\ b_4\ b_3\ b_2\ b_1\ b_0$	機能説明
ライトコマンド	62H	OUT	← GDCコマンドコード →	GDCコマンドについては，表4-9参照
ライトパラメータ	60H	OUT	← GDCパラメータ →	ライトコマンド命令を併用する
リードデータ	62H	IN	← GDCリードデータ →	ライトコマンド命令と併用する
ライト・モードレジスタ1	68H	OUT	← GDCモードデータ →	GDCモードレジスタの値を設定する
リードステータス	60H	IN	← GDCステータスデータ →	GDCの動作状態に関する情報を受け取る
CRTインタラプトリセット	64H	OUT	任意のデータ	CRTの割り込みをリセットする
ライト・ボーダーカラー	6CH	OUT	0 G R B 0 0 0 0	CRT画面のボーダー領域の色彩を指定する
ライトモードレジスタ2	6AH	OUT	0 0 0 0 0 0 0 DT	DT=0(8色)／1(16色)モードの選択

＊GDCコマンドを実行することと，GDCコマンドをライトコマンド命令でI/Oポート62Hに出力することとは，同一の意味である．

てデータの入力動作を実行しなければならないものもあり，この場合には，リードデータ命令を併用します．

多数あるGDCコマンドを４種類(動作制御用・表示制御用・描画制御用・VRAM制御用)に分類して整理したものを**表4-9**に示しています．そこでは，ライトパラメータ命令やリードデータ命令を併用する必要のあるGDCコマンドについては，GDCパラメータとGDCリードデータのデータ形式も併記しています．

なお，GDCに対して出力したGDCコマンドコードやGDCパラメータは，いずれも１バイトデータとして，GDC内部のFIFO*バッファにいったん蓄えられて順次実行されます．ただし，FIFOバッファのメモリサイズは16バイトであり，これがオーバーフロー状態のときに，GDCに対して出力されたGDCコマンドコードやGDCパラメータは無効となりますから注意が必要です．FIFOバッファの状態を知るには，次に述べるリードステータス命令を使用します．

以上は，GDCの機能全般についての説明でしたが，特にT-GDCでは，４種類に分類したGDCコマンドの中で，描画制御用のものが使用不可能ですから注意して下さい．一方，G-GDCでは，すべてのGDCコマンドが使用できます．

②リードステータス命令

リードステータス命令は，GDCの動作状態を示しているGDCステータスレジスタの内容を読み出すための命令です．レジスタから読み出したGDCステータスデータの各ビットの意味する内容を**表4-6**に示します．

表4-6　GDCステータスデータの示す情報

ビット番号	各ビットの示す内容（ビットの値＝1の時）
b_0	T-VRAMからデータを読み出し可能
b_1	FIFOバッファが満杯状態（GDCコマンド受付不可能）
b_2	FIFOバッファが空白状態
b_3	描画動作中
b_4	DMA動作中（98では使用しない）
b_5	垂直同期信号（VSYNC）を発生中
b_6	水平同期信号（HBLANK）を発生中
b_7	ライトペン位置の検出動作完了

③ライトモードレジスタ命令

ライトモードレジスタ命令は，T-GDCモードレジスタの値を設定するための命令です．T-GDCモードレジスタの各ビットがモード切り替えのスイッチの機能をなしていて，T-GDCモードレジスタの設定値によってT-GDCの動作モードを設定しています．

ライトモードレジスタ１命令では，GDCモードデータの４ビットb_3 b_2 b_1 b_0で構成されるモードフリップフロップ（Mode F/F）によって各種の設定を行います．ライトモードレジスタ命令でGDCモードデータを出力することにより，GDCモードデータの３ビットb_3 b_2 b_1に設定されているMode F/Fのビット番号を選択し，さらにビットb_0の値を０または１を選択することで，**表4-7**に示したようなT-GDCの動作モードを設定します．

ライトモードレジスタ２命令は，８色モードと16色モードの選択を行う命令です．本来，このコマンドはグラフィックに関わるものであって，テキストには関係ありません．このことからも明らかなように，厳密に言えば，GDCはテキスト用，グラフィック用に完全に機能分担されてはいません．また，T-GDC，G-GDCをそれぞれM-GDC(マスタ)，S-GDC（スレーブ）と呼ぶこともあります．

表4-7　T-GDCの動作モードを設定する Mode F/F

GDCモード・データ								Mode F/F		T-GDCのモード状態	備考
b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0	ビット番号	値		
0	0	0	0	0	0	0	0	b_0	1	簡易グラフモード	アトリビュートデータのビットb_4=1の時にのみ有効．　表4-3参照．
				0	0	0	1		0	垂線表示モード	
				0	0	1	0	b_1	1	モノクログラフィックモード	
				0	0	1	1		0	カラーグラフィックモード	
				0	1	0	0	b_2	1	40字モード	テキスト画面の1行当りの文字数
				0	1	0	1		0	80字モード	
				0	1	1	0	b_3	1	7×13ドットモード	文字フォントのサイズ
				0	1	1	1		0	6×8ドットモード	
				1	0	0	0	b_4	1	高解像度ディスプレイを200本モードで使用	グラフィック画面を構成する総ドット数
				1	0	0	1		0	高解像度400本または標準200本で使用	
				1	0	1	0	b_5	1	ビットマップモード	K-CG（漢字キャラクタジェネレータ）へのアクセス方式
				1	0	1	1		0	コードアクセスモード	
				1	1	0	0	b_6	1	書き込み許可モード	不揮発メモリへの書き込みの制御
				1	1	0	1		0	書き込み禁止モード	
				1	1	1	0	b_7	1	表示モード	画面表示制御
0	0	0	0	1	1	1	1		0	非表示モード	

（表の見方・例）GDCモードデータとして，0000 0101 を出力すれば，T-GDCモードレジスタのビットb_2が1にセットされ，T-GDCは40字モードに設定される．

3.2 G-GDCのI/O制御命令

G-GDCの機能を選択・設定するためのI／O制御命令について説明します．G-GDCの制御用に割り当てられているI／Oポートの種類は8種類あり，そのアドレスはA0H, A2H, A4H, A6H, A8H, AAH, ACH, AEHです．このI／Oポートを介して制御データを入出力することによって，G-GDCの制御を行っています．G-GDCのI／O制御命令を**表4-8**にまとめて示します．表には，各I／O制御命令の機能，使用するI／Oポートアドレス，および制御データの形式を示しています．

表4-8に示したI／O制御命令の名前について，より詳しく説明していきます．

①ライトコマンド命令，ライトパラメータ命令，リードデータ命令

この3種類の命令については，すでにT-GDCのI／O制御命令のところで説明したので，説明を省略します．**3.1①**を参照して下さい．

②リードステータス命令

この命令については，すでにT-GDCのI／O制御命令のところで説明したので，説明を省略します．**3.1②**を参照して下さい．

③表示画面選択命令，描画画面選択命令

G-VRAMのメモリマップの項（2.1(2)）で，G-VRAMがG-VRAM(1)とG-VRAM(2)の2組から構成されていることについて述べました．表示画面，描画画面選択命令は，2組あるG-VRAMのいずれを対象としてデータを入出力するかを選択するための命令です．この命令で出力する制御データの最下位ビットであるLSBの値SWが，選択のためのスイッチの機能を果しています．これは，**表4-2**でもすでに説明しました．

④ライトパレットレジスタA(B,C,D)命令

ライトパレットレジスタA命令は，パレットレジスタAにデータを格納するための命令です．同様に，ライトパレットレジスタB，C，D命令は，それぞれパレットレジスタB，C，Dにデータを格納するための命令です．この4つの命令を用いて，パレットレジスタに色（カラーモード時）または画面合成（モノクロモード時）を指定するデータを設定します．

表4-8　G-GDCのI/O制御命令

I/O制御命令	I/Oポートアドレス	I/O	制御データ b7	b6	b5	b4	b3	b2	b1	b0	機能説明
ライトコマンド	A2H	OUT	← GDCコマンドコード →								（注）GDCコマンドについては，表4-9参照．
ライトパラメータ	A0H	OUT	← GDCパラメータ →								ライトコマンド命令と併用する．
リードデータ	A2H	IN	← GDCリードデータ →								ライトコマンド命令と併用する．
リードステータス	A0H	IN	← GDCステータスフラグ →								GDCの動作状態に関する情報を受け取る．
表示画面選択	A4H	OUT	0	0	0	0	0	0	0	SW	G-VRAMの選択（画面表示時） SW＝0→G-VRAM(1)が選択される SW＝1→G-VRAM(2)が選択される
描画画面選択	A6H	OUT	0	0	0	0	0	0	0	SW	G-VRAMの選択（描画時） SW＝0→G-VRAM(1)が選択される SW＝1→G-VRAM(2)が選択される
ライトパレットレジスタD	A8H	OUT	0	G	R	B	0	G	R	B	上位(下位)4ビット→カラーパレット3(7)のカラーコード
			0	0	0	0	i	r	g	b	16種類のカラーパレットを選択するアドレスデータ
ライトパレットレジスタC	AAH	OUT	0	R	G	B	0	R	G	B	上位(下位)4ビット→カラーパレット1(5)のカラーコード
			0	0	0	0	G_3	G_2	G_1	G_0	カラーパレットの緑の強さを16階調で指定するデータ
ライトパレットレジスタB	ACH	OUT	0	G	R	B	0	G	R	B	上位(下位)4ビット→カラーパレット2(6)のカラーコード
			0	0	0	0	R_3	R_2	R_1	R_0	カラーパレットの赤の強さを16階調で指定するデータ
ライトパレットレジスタA	AEH	OUT	0	G	R	B	0	R	G	B	上位(下位)4ビット→カラーパレット0(4)のカラーコード
			0	0	0	0	B_3	B_2	B_1	B_0	カラーパレットの青の強さを16階調で指定するデータ

注）ライトパレットレジスタA（B，C，D）の制御データについて，上段は８色モード，下段は16モードに対応する．

注）カラーパレットのデータ
- ８色モード時　G R B　３ビット（$2^3=8$色）
- 16色モード時　G_3 G_2 G_1 G_0 R_3 R_2 R_1 R_0 B_3 B_2 B_1 B_0　12ビット

グラフィック時におけるCRTの表示モードには，**表4-1**に示したように８種類ありますが，各パレットレジスタＡ，Ｂ，Ｃ，Ｄの設定値の意味が表示モードによって異なるので，それぞれのモードの場合に分けて説明します．まず，カラーモードとモノクロモードに分類します．

(ｉ) カラーモード

(a) ８色グラフィックモード（標準グラフィックモード）

８色モード時には，パレットレジスタＡの設定値は，パレット番号０と１の色彩を定義するRGBコード（カラーコード）としての意味を持ちます．同様に，パレットレジスタＢでパレット番号２と３，パレットレジスタＣでパレット番号４と５，パレットレジスタＤでパレット番号６と７のRGBコード（カラーコード）を定義します．

(b) 16色グラフィックモード（拡張グラフィックモード）

16色モードでは，カラーパレットは16個あり，それぞれに0H～FHのパレット番号が与えられています．パレットレジスタＤのデータでまずパレット番号を指定します．次に，そのパレットの色彩をパレットレジスタＡ，Ｂ，Ｃで設定します．パレットレジスタＡのデータはＢ(青)の強度を2^4＝16階調で指定します．同様に，パレットレジスタＢ，ＣはそれぞれＲ（赤），Ｇ（緑）の強度を16階調で指定します．以上の手続きによって，16個のカラーパレットの色彩を設定します．したがって，１つのカラーパレットの設定データは，下記のように12ビットになります．

MSB　LSB

G_3	G_2	G_1	G_0	R_3	R_2	R_1	R_0	B_3	B_2	B_1	B_0

(ii) モノクロモード

(a) ８色グラフィックモード（標準グラフィックモード）

８色モードのときには，拡張G-VRAMは無駄になります．したがって，Ｂプレーン，Ｒプレーン，Ｇプレーンの３つのプレーンが有効です．モノクロモードでは，３つのプレーンの論理和を取ることによって，最大３画面までの画面合成ができます．どのプレーンを有効にするかは，パレットレジスタへのデータのセットの仕方で指定します．画面合成とパレットレジスタの関係を**図4-10**①に示します．

パレット番号は，ｂｒｇの３ビットで表されます．画面の１つのドットに対応する各プレーン（Ｇ，Ｒ，Ｂ）のビットデータがそれぞれｂ，ｒ，ｇである

第４章 グラフィックス

ときに，そのドットの描画に関しては番号ｂｒｇのパレット番号のデータが参照されます．したがって，モノクロ400モードの画面PB（Bプレーン）とPG（Gプレーン）を合成表示したければｂ＝１またはｇ＝１のときに参照されるパレット番号１，３，４，５，６，７のデータを7Hに，それ以外のパレット番号のデータを0Hにします．

図4-10　モノクロモード時における画面合成コード

①画面合成コード（８色モード）

画面合成			パレットレジスタの設定値							
PB	PR	PG	0	1	2	3	4	5	6	7
×	×	×	0	0	0	0	0	0	0	0
×	×	○	0	7	0	7	0	7	0	7
×	○	×	0	0	7	7	0	0	7	7
○	×	×	0	0	0	0	7	7	7	7
×	○	○	0	7	7	7	0	7	7	7
○	×	○	0	7	0	7	7	7	7	7
○	○	×	0	0	7	7	7	7	7	7
○	○	○	0	7	7	7	7	7	7	7

②画面合成コード（16色モード）

画面合成				パレットレジスタの設定値															
PB	PR	PG	PI	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
×	×	×	×	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
×	×	×	○	0	0	0	0	0	0	0	0	F	F	F	F	F	F	F	F
×	×	○	×	0	0	0	0	F	F	F	F	0	0	0	0	F	F	F	F
×	×	○	○	0	0	0	0	F	F	F	F	F	F	F	F	F	F	F	F
×	○	×	×	0	0	F	F	0	0	F	F	0	0	F	F	0	0	F	F
×	○	×	○	0	0	F	F	0	0	F	F	F	F	F	F	F	F	F	F
×	○	○	×	0	0	F	F	F	F	F	F	0	0	F	F	F	F	F	F
×	○	○	○	0	0	F	F	F	F	F	F	F	F	F	F	F	F	F	F
○	×	×	×	0	F	0	F	0	F	0	F	0	F	0	F	0	F	0	F
○	×	×	○	0	F	0	F	0	F	0	F	F	F	F	F	F	F	F	F
○	×	○	×	0	F	0	F	F	F	F	F	0	F	0	F	F	F	F	F
○	×	○	○	0	F	0	F	F	F	F	F	F	F	F	F	F	F	F	F
○	○	×	×	0	F	F	F	0	F	F	F	0	F	F	F	0	F	F	F
○	○	×	○	0	F	F	F	0	F	F	F	F	F	F	F	F	F	F	F
○	○	○	×	0	F	F	F	F	F	F	F	0	F	F	F	F	F	F	F
○	○	○	○	0	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F

注）パレット番号の設定値は12ビットであり，図中の0，Fはそれぞれ000H，FFFHの省略表現である。

(b)16色グラフィックモード（拡張グラフィックモード）

16色モードでは，Bプレーン，Rプレーン，Gプレーン，Iプレーンの4つのプレーンが有効です．モノクロモードでは4つの論理和を取ることによって最大4画面までの画面合成ができます．どのプレーンを有効にするかは，パレットレジスタへのデータのセットの仕方で指定できます．画面合成とパレットレジスタの関係を図4-10②に示します．

パレット番号は，ｉｇｒｂの4ビットで表されます．画面の1つのドットに対する各プレーン（I，G，R，B）のビットデータがそれぞれｉ，ｇ，ｒ，ｂであるときに，そのドットの描画に関しては番号 ｉｒｇｂ のパレット番号のデータが参照されます．したがって，モノクロ400モードの画面PI（Iプレーン）とPR（Rプレーン）を合成表示したければｉ＝1またはｒ＝1のときに参照されるパレット番号2，3，6，7，8，9，A，B，C，D，E，Fのデータを最大値FFFFH，それ以外のパレット番号のデータを最小値0000Hにします．

ただし，モノクロ200モードの場合には，各プレーンを2等分して使用しているので，例えばBプレーンを指定しただけでは，PB1，PB2のどちらかまでは決まりません．

これを指定するのが，図4-11に示したGDCパラメータの1つであるSADです．SADはG-VRAM上の表示開始GDCアドレスです．グラフィックモード時には，SADは各プレーン（B，R，G，I）に共通に使われ，各プレーンの先頭アドレスからのオフセットで表現されています．したがって，各プレーンの前半領域に対する画面（PB1，PR2，PG1，PI1）についての合成を行う場合には，SADを0000Hにし，後半領域に対する画面（PB2，PR2，PG2，PI2）についてはSADを1F40Hに設定します．

参考のために，それぞれの画面モードの設定に必要なGDCパラメータをGDCモードレジスタの値を図4-11に示しています．

図4-11　画面モードの指定に必要な設定項目

	画面構成	各モードに必要な設定項目				
		GDCモードレジスタ		GDCパラメータ		
		D_1	D_4	L/F	L/R	SAD
カラー400モード	Pb + Pr + Pg + Pi	0 (カラー)	0 (640×400)	400	1	0000H
カラー200モード	Pb1 + Pr1 + Pg1 + Pi1		1 (640×200)	200	2	0000H
	Pb2 + Pr2 + Pg2 + Pi2					1F40H

	画面構成	各モードに必要な設定項目				
		GDCモードレジスタ		GDCパラメータ		
		D_1	D_4	L/F	L/R	SAD
モノクロ400モード	PB	1 (モノクロ)	0 (640×400)	400	1	0000H
	PR					
	PG					
	PI					
モノクロ200モード	PB1		0 (640×200)	200	2	0000H
	PR1					
	PG1					
	PI1					
	PB2					1F40H
	PR2					
	PG2					
	PI2					

GDCモードレジスタについては表4-7参照.

GDCパラメータについては図4-12参照.

L/F：1画面当りのライン数
L/R：1ドットに対応するライン
SAD：VRAM上の表示開始GDCアドレス

画面構成の名称については図4-2参照.

（注）上記設定値は，ＣＲＴとして専用高解像度カラーＣＲＴが接続されていることを前提としている．標準ＣＲＴでは640×400ドットのモードに対応できない．

≡3.3≡ GDCの制御用サンプルプログラム

GDCの持つ機能をより具体的に理解していくために,これまで述べてきたGDCのI／O制御命令を用いたサンプルプログラムを示して，概説します．サンプルプログラムとして，カーソル形式の変更に関するもの，画面スクロールに関するもの，拡大表示に関するもの，ユーザ定義文字の描画に関するもの，以上4種類を示します．

ここでは，GDCの持つ機能と，その機能を引き出すためのI／O制御命令の活用法の基本とを理解することを目的としました．したがって，できるだけ単純なプログラムだけにとどめていますので,基本が理解できたら,読者自らI／O制御命令をどんどん活用して，ユニークな画面表示に挑戦していって下さい．

(1)カーソル形式の変更

ここでは，T-GDCのCSRFORMコマンドを用いて，カーソルの大きさとカーソルの点滅速度を変更するプログラムについて説明します．プログラムリストを**リスト4-1**に示します．**表4-5**に示したように，T-GDCのライトコマンド命令で使用するI／Oポートアドレスは62H，ライトパラメータ命令で使用するI／Oポートアドレスは60Hです．

160行でCSRFORMコマンドをT-GDCに出力し，170行～190行でGDCパラメータを1バイトずつ3回に分けて出力しています．各パラメータの意味については，**表4-9(2)**GDCコマンド［表示制御用］を参照して下さい．

リスト 4-1
GDCのI/O制御命令を用いたサンプルプログラム
(CSRFORMコマンドによるカーソル形式の変更)

```
1 'save "CURSOR",a
2 '
3 '     Cursor form set    V1.1
4 '     MS-DOS N88BASIC OK
100 WIDTH 80
101 *CSTIN : INPUT "Start lines   (0-15)",CST : IF CST>15 THEN *CSTIN
110 *CFIIN : INPUT "finish lines  (0-15)",CFI : IF CFI>15 THEN *CFIIN
120 *CSIN  : INPUT "Cursor  (ON=1/OFF=0)",CS  : IF CS>1   THEN *CSIN
130 *BDIN  : INPUT "Blinking(ON=1/OFF=0)",BD  : IF BD>1   THEN *BDIN
140 *BLIN  : INPUT "Blink Rate    (0-31)",BL  : IF BL>31  THEN *BLIN
150 BLH=(B ¥ 4)AND 7 : BLL=B MOD 4 : BD=1-BD
155 LINE INPUT WAIT 1,"設定したカーソル状態→";A$
160 OUT &H62,&H4B                'T-GDC CSRFORM command
170 OUT &H60,CS*&H80  + 15
180 OUT &H60,BLL*&H40 + BD*&H20 + CST
190 OUT &H60,CFI*&H8  + BLH
210 GOTO 210
220 END
```

⑵グラフィック画面の上下左右スクロール

ここでは，G-GDCのSCROLLコマンドを用いて，グラフィック画面を上下左右にスクロールさせるプログラムについて説明します．このコマンドを使用することによって，上下方向には１ドット単位で，左右方向には16ドット単位でスクロールさせることができます．プログラムリストを**リスト4-2**に示します．

それでは，実際の方法について少し触れてみます．SCROLLコマンドのそもそもの機能は，CRTに表示するVRAMの範囲を決定するためのコマンドです．

それを用いて１画面ごとにVRAM上の表示範囲を少しずつずらすことによって，あたかもスクロールしているように見えます．

この原理について，**図4-12**にも示しています．

リスト 4-2
GDCのI/O制御命令を用いたサンプルプログラム

（SCROLLコマンドを用いたグラフィック画面の上下左右スクロール）

```
1 'save "G-SCROLL",a
2 '
3 '  Graphic Screen Display Area Scroll Sample Program
4 '  MS-DOS N88BASIC OK , VX NG
100 SCREEN 3,0,0,1 : CLS 3
110 CONSOLE ,,0
120 CIRCLE(200,200),100,4
130 K$=INPUT$(1)
140 IF K$="4" THEN L=L+1  : GOTO 200
150 IF K$="8" THEN L=L+80 : GOTO 200
160 IF K$="6" THEN L=L-1  : GOTO 200
170 IF K$="2" THEN L=L-80 : GOTO 200
180 GOTO 130
190 '
200 IF ABS(L)>=&H3E80 THEN L=SGN(L)*(ABS(L)-&H3E80)
210 IF L<0 THEN RL=&H3E80+L ELSE RL=L
220 SL=&HC000+RL
230 SAD1=SL : SAD2=&HC000
240 IF L>=0 THEN              SL1=400-INT(L/40) : SL2=INT(L/40)
    ELSE SAD2=SAD2+L : SL1=INT(L/-40) : SL2=400-INT(L/40)
250 SAD1$=RIGHT$("000"+HEX$(SAD1),4)
255 LIN1$=RIGHT$("000"+HEX$(SL1),4)
260 SAD2$=RIGHT$("000"+HEX$(SAD2),4)
265 LIN2$=RIGHT$("000"+HEX$(SL2),4)
270 SAD1L=VAL("&h"+RIGHT$(SAD1$,2))
275 SL1L =VAL("&h"+RIGHT$(LIN1$,1))
280 SAD1H=VAL("&h"+LEFT$(SAD1$,2))
285 SL1H =VAL("&h"+MID$(LIN1$,1,3))
290 SAD2L=VAL("&h"+RIGHT$(SAD2$,2))
295 SL2L =VAL("&h"+RIGHT$(LIN2$,1))
300 SL2H =VAL("&h"+MID$(LIN2$,1,3))
305 SAD2H=VAL("&h"+LEFT$(SAD2$,2))
310 WAIT &HA0,&H20
320 OUT &HA2,&H70                          'SCROLL COMMAND
330 OUT &HA0,SAD1L                         'FACE1
340 OUT &HA0,SAD1H
350 OUT &HA0,SL1L*&H10
360 OUT &HA0,SL1H
370 OUT &HA0,SAD2L                         'FACE2
380 OUT &HA0,SAD2H
390 OUT &HA0,SL2L*&H10
400 OUT &HA0,SL2H
410 GOTO 130
420 END
```

図4-12　スクロール原理説明

⑶グラフィック画面の拡大表示

ここでは，G-GDCのCSRFORMコマンドを用いて，グラフィック画面を拡大表示させるプログラムについて説明します．拡大表示させるためのコマンドとしてZOOMコマンドがありますが，この場合の最大倍率は15倍です．しかも，描画時のみ可能という制限がつきます．これに対して，CSRFORMコマンドを使えば最大倍率が32倍です．

プログラムのリストを**リスト4－3**に示します．

また，この原理を説明すれば，CSRFORMコマンドのパラメータL／R，つまり縦と横の長さの比を変更することによって，拡大を行っています．

リスト 4-3
GDCのI/O制御命令を用いたサンプルプログラム
(CSRFORMコマンドを用いたグラフィック画面の拡大表示)

```
1 'save "ZOOM",a
2 '
3 '     Zooming sample program V1.1
4 '     MS-DOS N88BASIC OK , VX NG
100 SCREEN 3,0,0,1 : CLS 3
110 'Init Graph
120 FOR I=0 TO 639
130   LINE(I,0)-(639-I,399),I MOD 7+1
140 NEXT
150 'Zoom Up
160 FOR ZR=0 TO 31
170   OUT &HA2,&H4B             'G-GDC CSRFORM command
180   OUT &HA0,ZR               'Zoom Rate(L/R)
190   LINE INPUT WAIT 1,A$      'Wait
200 NEXT ZR
210 'Zoom Down
220 FOR ZR=31 TO 0 STEP -1
230   OUT &HA2,&H4B
240   OUT &HA0,ZR
250   LINE INPUT WAIT 1,A$
260 NEXT ZR
270 END
```

(4)ユーザ定義文字の描画

ここでは，G-GDCのVECTWコマンドを用いて，ユーザが定義した文字をグラフィック画面に表示させるプログラムについて説明します．

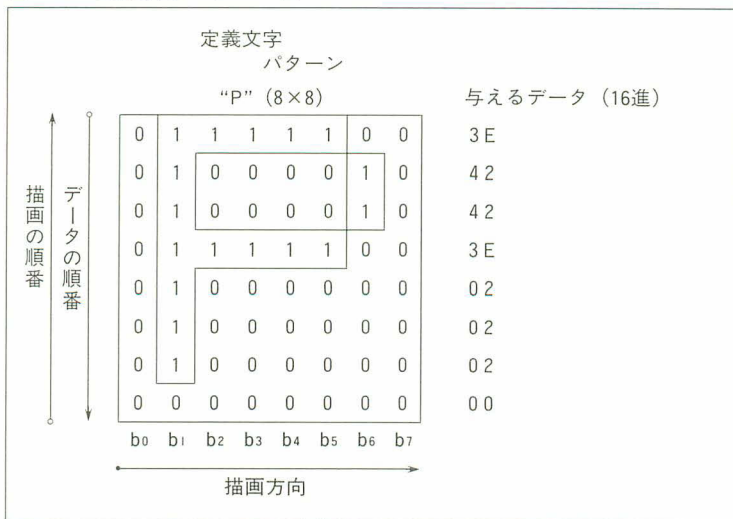
プログラムリストを**リスト4-4**に示します．

このプログラムについては，まず280行～310行でCSRWコマンドを用いて表示位置を決定して，320行～340行で描画方向を決定し，最後にTEXTEコマンドを用いて描画を開始します．

リスト 4-4
GDCのI/O制御命令を用いたサンプルプログラム
（VECTWコマンドを用いたユーザ定義文字のグラフィック描画）

```
1 'save "GDCCHR",a
2 '
3 ' G-GDC "Graphic Character Writing" Sample Program V1.1
4 ' MS-DOS N88BASIC OK
100 SCREEN 3,0,0,1 : CLS 3 : CONSOLE ,,0 : LOCATE ,,0
110 ON STOP GOSUB *ABORT : STOP ON
120 FOR K=0 TO 6
130   CLS 2 : RESTORE
140   X=100 : Y=80 : DX=50 : DY=50 : ZOOM=4 : COL=K MOD 3+1
150   FOR J=0 TO 5
160     DIR=K
170     OUT &HA2,&H46           'ZOOM command
180     OUT &HA0,ZOOM
190     OUT &HA2,&H78           'Character pattern Set command
200     FOR I=0 TO 7
210       READ A$ : OUT &HA0,VAL("&h"+A$)
220     NEXT
230     OUT &HA2,&H23           'Write command
240     EAD =X ¥ 16 + 40*Y + &H4000*COL
250     EADH=INT(EAD / 256)
260     EADL=EAD - EADH*256
270     DAD =X MOD 16
280     OUT &HA2,&H49           'CSRW  command
290     OUT &HA0,EADL
300     OUT &HA0,EADH
310     OUT &HA0,DAD*16
320     OUT &HA2,&H4C           'VECTW command
330     OUT &HA0,DIR+16
340     OUT &HA0,7
350     OUT &HA2,&H68           'TEXTE command
360     X=X+DX : Y=Y+DY
370     LINE INPUT WAIT 1,A$    'Wait
380   NEXT
390 NEXT
400 *ABORT
410 LOCATE ,,1
420 END
430 ' Character Pattern  (8*8)
440 DATA 00,40,40,40,7C,42,42,7C
450 DATA 00,1C,22,40,40,40,22,1C
460 DATA 00,38,04,02,3E,42,42,3C
470 DATA 00,3C,42,42,3C,42,42,3C
480 DATA 00,3C,42,62,5A,46,42,3C
490 DATA 00,3E,08,08,08,28,18,08
```

図4-13　定義文字パターン図

0	1	1	1	1	1	0	0	3E
0	1	0	0	0	0	1	0	42
0	1	0	0	0	0	1	0	42
0	1	1	1	1	1	0	0	3E
0	1	0	0	0	0	0	0	02
0	1	0	0	0	0	0	0	02
0	1	0	0	0	0	0	0	02
0	0	0	0	0	0	0	0	00
b_0	b_1	b_2	b_3	b_4	b_5	b_6	b_7	

さて，肝心の定義文字の登録ですが，これは最後に並べてあるデータおよび260行～280行でGDCに対して登録しています．なお，データの修正に関しては，**図4-13**を参照して下さい．

3.4 GDCコマンド一覧

GDCのI／O制御命令については，すでに3.1，3.2で説明しました．そこでは，I／O制御命令を実行するためには，指定されたI／Oポートアドレスに向けて，GDCコマンドコードやGDCパラメータを選択する必要があることを述べました．そして，GDCコマンドには，動作制御用，表示制御用，描画制御用，VRAM制御用の4種類あること，T-GDCでは描画制御用コマンドが使用できないことも述べました．GDCコマンドコードの一覧を，それぞれの種類に分けて，**表4-9**にまとめておきます．

表4-9(1)　GDCコマンド(動作制御用)

GDCコマンド	GDCコマンドコード　GDCパラメータ　GDCリードデータ(注1)										機能説明
	b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0	タイプ	コマンドコード値	
MASTER/SLAVE	0	1	1	0	1	1	1	M	C	6FH/6EH	T-GDC／G-GDCの選択 {M＝0→G-GDC, M＝1→T-GDC}
RESET	0	0	0	0	0	0	0	0	C	00H	GDCの初期化
SYNC	0	0	0	0	1	1	1	DE	C	0FH/0EH	動作モードの選択 同期信号の設定 {DE＝0→表示停止, DE＝1→表示開始}
	0	0	CHR	F	I	D	G	S	P_1		表示モード，動作モードの設定
	←C/R→								P_2		1行当りの桁数設定，(テキスト画面に対して)
	←VS_L→			←HS→					P_3		水平・垂直同期信号の設定 非表示領域の設定 1画面当りのライン数設定 (P_3～P_8)
	←HFP→						←VS_H→		P_4		
	0	0	←HBP→						P_5		
	0	0	←VFP→						P_6		
	←L/F_L→								P_7		
	←VBP→						←L/F_H→		P_8		

(注)タイプ欄の記号
- C：GDCコマンドコード
- Pi：GDCパラメータ
- Ri：GDCリードデータ

表中の記号の説明

表中の記号	機能の説明
CHR	テキスト画面において 表示モードと描画モードの選択　{CHR＝0→描画モード CHR＝1→表示モード
G	グラフィック画面において 表示モードと描画モードの選択　{G＝0→描画モード G＝1→表示モード
F	描画タイミング　{F＝0→フラッシュ描画 F＝1→フラッシュレス描画
I	インターレス走査の有無　{I＝0→無　T-VRAM I＝1→有　G-VRAM
S	ラインカウンタや表示アドレスの 進み方の形態を設定．通常S＝1
D	VRAMの素子種別に応じて， リフレッシュ動作の必要性の有無を設定　{D＝0→不要 D＝1→要　(通常時)
C/R	1行当りの文字数（テキスト画面）
VS	垂直同期信号の幅．　VS＝08H（通常時）
HS	水平同期信号の幅．　HS＝07H（通常時）
HFP	CRTの右側部の非表示区間
VFP	CRTの下側部の非表示区間
HBP	CRTの左側部の非表示区間
VBP	CRTの上側部の非表示区間
L/F	1画面当りのライン数

表4-9(2)　GDCコマンド(表示制御用)

GDCコマンド	GDCコマンドコード　GDCパラメータ　GDCリードデータ										機能説明
	b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0	タイプ	コマンドコード値	
START	0	1	1	0	1	0	1	1	C	69H	表示開始
STOP	0	0	0	0	1	1	0	0	C	0CH	表示停止
ZOOM	0	1	0	0	0	1	1	0	C	46H	拡大係数の設定
	×	×	×	×	←ZW→				P		グラフィック文字描画時の拡大係数
SCROLL	0	1	1	1	←RA*→				C	70H	画面の複数分割を設定。{G-GDC：2領域に分割可能 / T-GDC：4領域に分割可能}
	←$SAD1_L$→								P_1		第1領域の設定
	←$SAD1_H$→								P_2		
	←$SL1_L$→				0	0	0	0	P_3		
	DAD	IM	←$SL1_H$→						P_4		
	←$SAD2_L$→								P_5		第2領域の設定 (第3領域，第4領域の設定は以下同様)
	←$SAD2_H$→								P_6		
	←$SL2_L$→				0	0	0	0	P_7		
	DAD	IM	←$SL2_H$→						P_8		
CSRFORM	0	1	0	0	1	0	1	1	C	4BH	カーソル形式の設定
	CS	0	0	←L/R→					P_1		カーソル表示のON，OFF．1行当りの桁数
	←BL_L→		BD	←CST→					P_2		カーソル点滅周期，etc.
	←CFI→					←BL_H→			P_3		カーソル表示終了ライン番号
PITCH	0	1	0	0	0	1	1	1	C	47H	VRAMの構成の設定
	←P→								P		VRAMの横方向のアドレス数
LPEN	1	1	0	0	0	0	0	0	C	C0H	ライトペンアドレスの読み出し
	←LAD_L→								R_1		ライトペンアドレス
	←LAD_H→								R_2		
	×	×	×	×	×	×	←LAD_H→		R_3		

＊RA（内蔵RAMのアドレス）
　パラメータの書き込み後，自動インクリメントする．P_1～P_{16}がアドレス0H～FHに相当。

表中の記号の説明

図中の記号	機能説明
SAD1 SAD2	VRAM上の表示開始GDCアドレス
SL1 SL2	各分割画面の表示領域のライン数
DAD	表示アドレスの増加状態の指定　{DAD＝0 → DAD＝DAD＋1 DAD＝1 → DAD＝DAD＋2
IM	表示アドレスを増加させるタイミングの設定　{IM ＝0 → テキスト画面に対して IM ＝1 → グラフィック画面に対して
CS	カーソル表示のON，OFF　{CS ＝0 → OFF CS ＝1 → ON
L/R	{テキスト画面に対しては，1行当りのライン数を設定する グラフィック画面に対しては，1ドット当りに対応するライン数を設定する．通常1又は2.
BL	カーソル点滅周期の設定
BD	カーソル点滅のON，OFF　{BD ＝0 → 点滅 BD ＝1 → 常時点灯
CST	カーソル表示開始ライン番号
CFI	カーソル表示終了ライン番号

表4-9(3) GDCコマンド(描画制御用)

GDCコマンド	b7	b6	b5	b4	b3	b2	b1	b0	タイプ	コマンドコード値
	GDCコマンドコード GDCパラメータ GDCリードデータ									
VECTW	0	1	0	0	1	1	0	0	C	4CH
	SL	R	C	T	L	← DIR →			P_1	
	← DC_L →								P_2	
	0	DGD	← DC_H →						P_3	
	← D_L →								P_4	
	×	×	← D_H →						P_5	
	← $D2_L$ →								P_6	
	×	×	← $D2_H$ →						P_7	
	← $D1_L$ →								P_8	
	×	×	← DI_H →						P_9	
	← DM_L →								P_{10}	
	×	×	← DM_H →						P_{11}	
VECTE	0	1	1	0	1	1	0	0	C	6CH
TEXTW	0	1	1	1	1	← RA →			C	78H
	← TX8(PTN_L) →								P_1	
	← TX7(PTN_H) →								P_2	
	← TX6 →								P_3	
	← TX5 →								P_4	
	← TX4 →								P_5	
	← TX3 →								P_6	
	← TX2 →								P_7	
	← TX1 →								P_8	
TEXTE	0	1	1	0	1	0	0	0	C	68H
CSRW	0	1	0	0	1	0	0	1	C	49H
	← EAD_L →								P_1	
	← EAD_H →								P_2	
	← dAD →				0	0	← EAD_H →		P_3	

機能説明

VECTW: 直線, 四辺形, 円を描画するためのパラメータ設定

	DC	D	D2	DI	DM
初期値	0	8	8	−1	−1
直線	|ΔX|	2|ΔY|−|ΔX|	2|ΔY|−2|ΔX|	2|ΔY|	
円・弧	N	$r-1$	$2(r-1)$	−1	M
四辺形	3	A	B	−1	A

ΔX:X座標変位　ΔY:Y座標変位　r:半径　N:描画総ドット数
M:マスキング・ドット数　A:第1辺のドット数　B:第2辺のドット数
注　Y軸方向に±45°の領域に対して直線を描画する場合には
　ΔXとΔYの値を交換します.

VECTE: (記載なし)

TEXTW: 描画する線種および文字パターンの設定

線種の設定　　　→ PTN_L, PTN_H
文字パターンの設定→ TX1〜TX8

描画方向
LSB ――――→ MSB
(TX1 ↓ TX8)

TEXTE: 定義文字の描画開始

CSRW: 描画開始アドレスの設定
(G-GDCのとき, すべて設定
T-GDCのとき, P_1, P_2でEADのみ設定)

CSRR	1 1 1 0 0 0 0 0	C	描画開始アドレス設定値の読み出し
	←EAD$_L$→	R_1	(描画終了時に，次に描画するアドレス内容に
	←EAD$_M$→	R_2	更新されているので、連続描画が可能)
	X X X X X X ←EAD$_H$→	R_3	
	←dAD$_L$→	R_4	
	←dAD$_H$→	R_5	
MASK	0 1 0 0 1 0 1 0	C	文字描画時，ドット単位のマスクを設定
	←MASK$_L$→	P_1	
	←MASK$_H$→	P_2	

表中の記号の説明

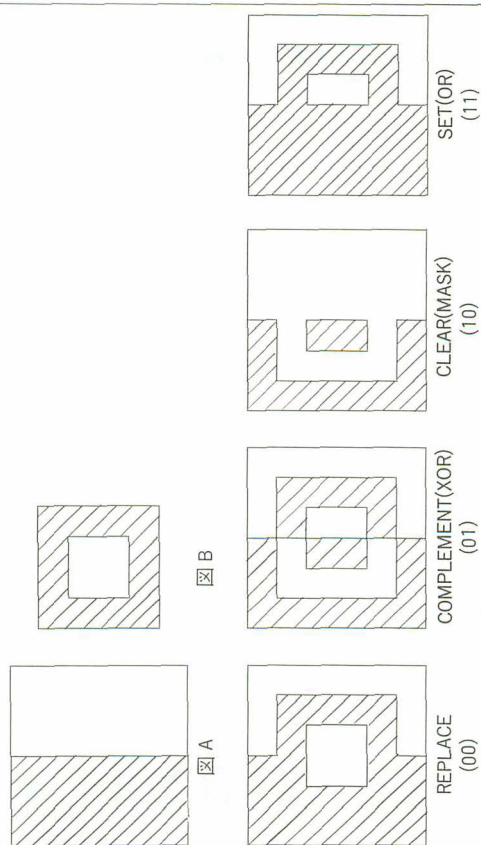
表中の記号	機能説明
SL	描画種類 {SL＝0 → グラフィック文字を指定しない SL＝1 → グラフィック文字を指定}
T	グラフィック文字の書体 {T＝0 → 通常 T＝1 → 斜体}
C，R，L	描画種類 {C＝0 C＝1 → 円} {R＝0 R＝1 → 四辺形} {L＝0 L＝1 → 直線}
DIR	描画方向（DIR＝0～7)，0のとき下方向，増加につれ反時計まわり(第４章6.3(7)注2)
DGD	描画時のアドレスの進み方
DC，D，D1 D2，DM	描画パラメータ(VECTWコマンド参照)
PTN	線種データ（実線，破線 etc.）
TX1～TX8	グラフィック文字描画時に参照されるドット構成のデータ
EAD	描画開始ワードアドレス
dAD	描画開始ドットアドレス（dAD＝0～15）
MASK	ドットアドレスとマスキングに兼用しているレジスタの設定値

表4-9(4)　GDCコマンド(VRAM制御用)

GDCコマンド	GDCコマンドコード　GDCパラメータ　GDCリードデータ										機能説明
	b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0	タイプ	コマンドコード値	
WRITE 0	0	0	1	0	0	0	←MOD→		C	20H~23H	ワード単位で，データをVRAMに転送
	←CODE$_L$→								P_1		下位バイトデータ
	←CODE$_H$→								P_2		上位バイトデータ
WRITE 1	0	0	1	1	0	0	←MOD→		C	30H~33H	バイト単位で，下位バイトデータのみをVRAMに転送
	←CODE$_L$→								P		下位バイトデータ
WRITE 2	0	0	1	1	1	0	←MOD→		C	38H~3BH	バイト単位で，上位バイトデータのみをVRAMに転送
	←CODE$_H$→								P		上位バイトデータ
READ 0	1	0	1	0	0	0	←MOD→		C	A0H~A3H	ワード単位で，データをVRAMから読み出す
	←CODE$_L$→								R_1		下位バイトデータ
	←CODE$_H$→								R_2		上位バイトデータ
READ 1	1	0	1	1	0	0	←MOD→		C	B0H~B3H	バイト単位で，下位バイトデータのみをVRAMから読み出す
	←CODE$_L$→								R		下位バイトデータ
READ 2	1	0	1	1	1	0	←MOD→		C	B8H~BBH	バイト単位で，上位バイトデータのみをVRAMから読み出す
	←CODE$_H$→								R		上位バイトデータ

表中の記号の説明

表中の記号	機能説明
MOD	ドット修正モードの設定．VRAMに既に格納されているデータと，新しく書き込むデータとの演算方法を規定する． 具体的には，下図参照．　00→REPLACE　01→COMPLEMENT　10→CLEAR　11→SET
CODE	VRAMに書き込むデータ．あるいは，VRAMから読み出すデータ．複数も可．

ドット修正モードの説明
VRAMに既に格納されているデータに対応するビットパターンを図Aとする.
VRAMに新しく格納するデータに対応するビットパターンを図Bとする.
図Bに対応する新しいデータを、図Aに対応する状態にあるVRAMに，それぞれのドット修正モードで書き込んだ場合に得られるVRAMのデータ格納状態を以下に示す.
図A
図B
REPLACE
(00)
COMPLEMENT(XOR)
(01)
CLEAR(MASK)
(10)
SET(OR)
(11)

4 CRTC

CRTC*は，GDC**とともにCRT制御の中枢を占めている LSI（μPD52611）であり，特に，テキスト画面に文字を表示する場合の垂直方向のタイミング制御を行っています．CRTCの主な機能を列記してみると，

① CG***が出力するキャラクタパターンを画面上に表示する際のタイミング信号を生成する．

② アンダーラインを表示する際のタイミング信号を生成する．

③ スムーススクロール，つまり1ドット単位で上下スクロールさせる際のタイミング信号を生成する．

などがあります．

多数あるCRTCの機能のうち，いずれの機能もCRTCに対してI/O制御命令を与えることによって選択・設定できます．具体的には，CRTCの制御用に割り当てられているI/Oポートアドレスを介して，制御データを入出力することで命令を与えます．I/O制御命令についての詳細は，本節の(1)項で述べます．

また，CRTCの機能とその制御方法についての理解を深めるために，I/O制御命令を用いたサンプルプログラムを示して解説します．

4.1 CRTCのI/O制御命令

CRTCの機能を選択・設定するためのI/O制御命令について説明します．CRTCの制御用に割り当てられているI/Oポートの種類は6種類あり，そのアドレスは70H，72H，74H，76H，78H，7AHです．このI/Oポートを介して制御データを入出力することによって，CRTCの制御を行っています．CRTCのI/O制御命令を**表4-10**にまとめて示します．表には，各I/O制御命令の状態，使用するI/Oポートアドレス，および制御データの形式を示しています．

表4-10に示したI/O制御命令の各々について，より詳しく説明していきます．

---

*　CRTC＝CRT Controller

**　GDC＝Graphic Display Controller．本章3参照

***　CG＝Character Generator．本章5参照

表4-10　CRTCのI/O制御命令

I/O制御命令	I/Oポートアドレス	I/O	制御データ b_7 b_6 b_5 b_4 b_3 b_2 b_1 b_0	機能説明
ライトPL	70H	OUT	←　PL　→	ボディフェースの上端のライン番号を設定し，これを初期値として，ラインカウンタに格納する．
ライトBL	72H	OUT	←　BL　→	ボディフェースの下端のライン番号を設定する．
ライトCL	74H	OUT	←　CL　→	キャラクタフェースのライン数を設定する．
ライトSSL	76H	OUT	←　SSL　→	スクロールエリアの文字を上方にスクロールしているライン数
ライトSUR	78H	OUT	←　SUR　→	スクロールエリアの開始位置の行数を設定する．(2の補数表現)
ライトSDR	7AH	OUT	←　SDR　→	スクロールエリアの行数－1を設定する．

(1)ライトPL命令，ライトBL命令，ライトCL命令

この３種類の命令を説明する前に，まずボディフェース，キャラクタフェースについて説明しておきます．

テキスト画面が基本表示領域を単位にして分割され，それぞれの領域にテキストアドレスを割り当てて区別することについて，すでにT-VRAMの箇所で説明しました（図4-5①を参照）．この基本表示領域のことをボディフェースと呼びます．そして，ボディフェース内で文字を表現するために実質的に使用される領域をキャラクタフェースと呼びます．640×400ラインで構成される画面を20行モードのテキスト画面に設定すると，ボディフェースの垂直方向ラインは20です．この20本のラインに対するライン番号の割り付け方を図4-14に示します．

図の10進表現に着目して下さい．まず，キャラクタフェースの上端に相当するラインの番号を０とします．これを基準に整数値を順次割り当てています．次に，これを２進表現しますが，その際，５ビットを利用し，２の補数表現で負の数を表します．

以上で，前置きを終えて，次にライトPL命令について説明します．この命令は，データPL，つまりボディフェース上端のライン番号（図4-14で言えば１CH）をラインカウンタ*に格納するための命令です．

次に，ライトBL命令は，データBL，つまりボディフェース下端のライン番号（図4-14で言えば０FH）を所定のメモリに格納するための命令です．

＊　描画中のライン番号を指しているカウンタ

第4章 グラフィックス

そして，ライトCL命令は，データCL，つまりキャラクタフェースのライン数（**図4-14**で言えば０CH）を所定のメモリに格納するための命令です．

(2)ライトSSL命令，ライトSUR命令，ライトSDR命令

この３種類の命令を説明する前に，まずスクロールエリアを含む画面について説明しておきます．

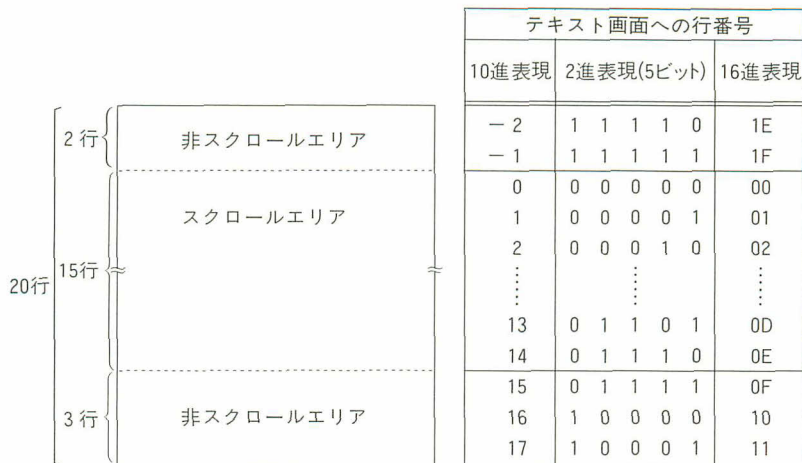
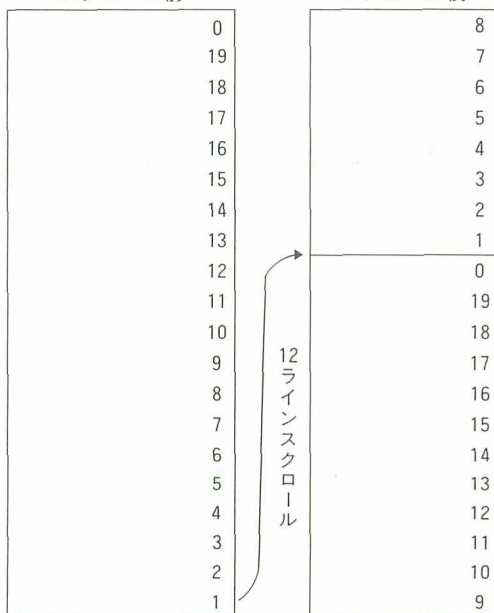
640×400ラインで構成される画面を，20行モードのテキスト画面に設定しているものとします．このとき，**図4-15①**に示すように，スクロールエリアを画面中に設定することを考えます．図中の表は，テキスト画面への行番号の割り

図4-14　ボディフェースに対するライン番号の割り当て

			ライン番号の割り当て			備考
			10進表現	2進表現(5ビット)	16進表現	(注)
20ライン	4ライン	ボディフェース	－4	1 1 1 0 0	1C	→PL
			－3	1 1 1 0 1	1D	
			－2	1 1 1 1 0	1E	
			－1	1 1 1 1 1	1F	
	12ライン	キャラクタフェース	0	0 0 0 0 0	00	→CL
			1	0 0 0 0 1	01	
			2	0 0 0 1 0	02	
			3	0 0 0 1 1	03	
			4	0 0 1 0 0	04	
			5	0 0 1 0 1	05	
			6	0 0 1 1 0	06	
			7	0 0 1 1 1	07	
			8	0 1 0 0 0	08	
			9	0 1 0 0 1	09	
			10	0 1 0 1 0	0A	
			11	0 1 0 1 1	0B	
	4ライン		12	0 1 1 0 0	0C	
			13	0 1 1 0 1	0D	
			14	0 1 1 1 0	0E	
			15	0 1 1 1 1	0F	→BL

（注）記号の使い方は表4-10中の制御データと対応しています。

図4-15　スクロールエリアを含むテキスト画面に対する行番号の割り当て

テキスト画面への行番号		
10進表現	2進表現(5ビット)	16進表現
－2	1 1 1 1 0	1E
－1	1 1 1 1 1	1F
0	0 0 0 0 0	00
1	0 0 0 0 1	01
2	0 0 0 1 0	02
⋮	⋮	⋮
13	0 1 1 0 1	0D
14	0 1 1 1 0	0E
15	0 1 1 1 1	0F
16	1 0 0 0 0	10
17	1 0 0 0 1	11

付けを示しています．図の10進表現に着目して下さい．まず，スクロールエリアの上端に相当する行の番号を0とします．これを基準に，整数値を順次割り当てていきます．次に，これを2進表現しますが，その際，5ビットを使用し，2の補数表現で負の数を表します．

一方，**図4-15②**は，スクロール量の定義を示しています．今の場合，1行は20ラインから構成されていて，スクロール前に比べて，スクロール後では基準位置が12ライン分だけ上方へスクロールしています．スクロール量は，0～19(20の剰余系）で表します．上方へスクロールする場合を，スクロール量の増加方向とします．

以上で前置きを終えて，次に，ライトSSL命令について説明します．この命令は，データSSL，つまりスクロールエリア内の文字が最初の基準位置からスクロールしたときのスクロール量（**図4-15②**で言えば12）を所定のメモリに格納するための命令です．

次に，ライトSUR命令は，データSUR，つまりテキスト画面上端の行番号（**図4-15①**で言えば1EH）を所定のメモリに格納するための命令です．

そして，ライトSDR命令は，データSDR，つまりスクロールエリア下端の行番号（**図4-15①**で言えば0EH）を所定のメモリに格納するための命令です．ライトSUR命令とライトSDR命令でスクロールエリアを指定しています．

4.2 CRTCのI/O制御命令を用いたサンプルプログラム

CRTCの持つ機能を，より具体的に理解するために，これまで述べてきたCRTCのI/O制御命令を用いたサンプルプログラムを**リスト4-5**に示し，概説します．

これは，画面をスムーススクロールさせるプログラムです．以下に，動作を簡単に説明します．まず，2つのレジスタSUR，SDRでスクロール位置を決定しています．そして，SSLの値を変えることにより，1ラインずつスクロールさせています．これで1キャラクタ分まで順次スクロールできますが，さらに1キャラクタを越えてスクロールを続行するためには，GDCにSCROLLコマンドを送出して，初期設定を変更する必要があります．

リスト 4-5
CRTCのI/O制御命令を用いた
サンプルプログラム

```
1 'save "T-SCROLL",a
2 '
3 '  Text Screen Smooth Scroll Sample Program V1.1
4 '  MS-DOS N88BASIC OK
100 ON STOP GOSUB *DONE : STOP ON
105 WIDTH 80,25 : CONSOLE ,,0,1
110 FOR I=1 TO 24
120   COLOR I MOD 7+1 : PRINT STRING$(80,CHR$(&H40+I));
130 NEXT
140 OUT &H78,&H1E        'SUR & SDR set
150 OUT &H7A,9
160 FOR RW=1 TO 20
170   OUT &H64,0         'CRT reset
180   FOR CO=0 TO 15
190     OUT &H76,CO      'SSL set
200     FOR W=0 TO 30 : NEXT 'wait
210   NEXT
220   SAD2 =80 + 80*RW          'Scroll area change
230   SAD2H=INT(SAD2 / 256)
240   SAD2L=SAD2 - SAD2H * 256
260   OUT &H62,&H70
270   OUT &H60,&H0              'No.1 face      AAA...A
280   OUT &H60,&H0              '                 ¦
290   OUT &H60,&HE0             '               BBB...B
300   OUT &H60,&H1
310   OUT &H60,SAD2L            'No.2 face      CCC...C
320   OUT &H60,SAD2H            '                 ¦      ~ LLL...L
330   OUT &H60,&H50             '               LLL...L     ¦
340   OUT &H60,&HA              '                          VVV...V
350   OUT &H60,&HC0             'No.3 face
360   OUT &H60,&H3              '               MMM...M
370   OUT &H60,&H30             '                 ¦
380   OUT &H60,&HC
390   OUT &H76,0
400 NEXT
410 *DONE
420 OUT &H78,0        'All reset
430 OUT &H7A,0
440 OUT &H62,&H70
450 OUT &H60,&H0
460 OUT &H60,&H0
470 OUT &H60,&H0
480 OUT &H60,&H28
490 OUT &H76,0
500 COLOR 7
510 END
```

5 CG

CG*は，ANK文字や日本字などのフォントパターンを発生するための回路です．CGを構成する主要素子は，文字のフォントパターンを記録してある多数のROMです．特に，ANK文字の表示に関するCGをANK-CG，日本字に関するCGをK-CGと呼びます．K-CGではROMに加えて外字用RAMも併用しているので，ここにユーザ独自の文字パターンを定義・登録することもできます．

CGの主な機能を列記します．

①ROM，あるいは外字用RAM上に登録されている文字パターンを読み出し，画面表示する

②ユーザ定義文字パターンを外字用RAM上に書き込んで登録する

CGの様々な機能は，CGに対して制御命令を与えることにより，選択・設定できます．具体的には，CGの制御用に割り当てられているI/Oポートを介して制御データを入出力することで命令を与えます．制御命令についての詳細は，5.1で述べます．

また，5.2では，CGの機能とその制御方法についての理解を深めるために，制御命令を用いたサンプルプログラムを示して解説します．

5.1 CGのI/O制御命令

CGの機能を選択・設定するためのI/O制御命令について説明します．CGの制御用に割り当てられているI/Oポートの種類は4種類あり，そのアドレスはA1H, A3H, A5H, A9Hです．このI/Oポートを介して制御データを入出力することにより，CGの制御を行っています．CGのI/O制御命令を**図4-16**にまとめて示します．図には，各I/O制御命令の機能，使用するI/Oポートアドレス，および制御データの形式を示しています．

図4-16に示したI/O制御命令の各々について，より詳しく説明していきます．

---

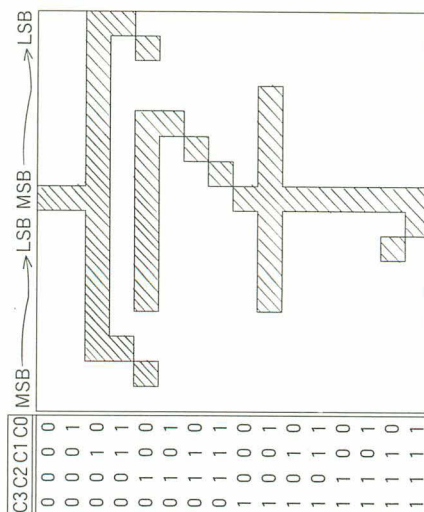
* CG＝character Generator

図4-16　CGのI/O制御命令

I/O制御命令	I/Oポート アドレス	I/O	制御データ b_7 b_6 b_5 b_4 b_3 b_2 b_1 b_0	機能説明
ライトコードH	A3H	OUT	←文字コード(上位バイト)→	JISコードの上位バイト－20Hの値を設定する．
ライトコードL	A1H	OUT	←文字コード(下位バイト)→	JISコードの下位バイトを設定する．
ライトカウンタ	A5H	OUT	× × L/R 0 C3 C2 C1 C0	文字のビットパターンの中から，対象とする位置を設定する．
リードパターン	A9H	IN	←ビットパターン→	メモリから，文字のビットパターンを読み出す．
ライトパターン	A9H	OUT	←ビットパターン→	メモリに文字のビットパターンを書き込む．(ユーザ文字の定義)

図中の記号	機能説明
L/R	右図に示した16×16ドットのパターンの左右領域のいずれを対象にするかを選択する．L/R=1→左，L/R=0→右
C0, C1 C2, C3	右図に示した16×16ドットのパターンの何段目を対象とするかを選択する．設定値を右図に示す．

MSB→LSB MSB→LSB

C3	C2	C1	C0
0	0	0	0
0	0	0	1
0	0	1	0
0	0	1	1
0	1	0	0
0	1	0	1
0	1	1	0
0	1	1	1
1	0	0	0
1	0	0	1
1	0	1	0
1	0	1	1
1	1	0	0
1	1	0	1
1	1	1	0
1	1	1	1

(1)ライトコードH命令，ライトコードL命令

この命令は，K-CG上のROM(または外字用RAM)から読み出したい文字のコードを設定するための命令です．あるいは，ユーザが独自に文字パターンをK-CGの外字用RAMに登録する場合に，その文字パターンに与える文字コードを設定する命令でもあります．2バイトの文字コードのうち，上位バイトをライトコードH命令で，下位バイトをライトコードLで設定します．

(2)ライトカウンタ命令

この命令は，16×16ビットの文字パターンを構成する32バイトのデータのうち，どの1バイトを対象として，データの読み出し，あるいは書き込みをなすかを設定するための命令です．

(3)リードパターン命令

この命令は，ライトカウンタの命令で指定している1バイトデータを読み出すための命令です．ライトカウンタ命令で，対象とするデータを順次変更しながら，この命令を実行すれば16×16ビットのパターン全部を読み出すことができます．

(4)ライトパターン命令

この命令は，リードパターン命令の逆操作を行う命令であり，16×16ビットのパターンを構成する32バイトのデータを，順次K-CGの外字用RAMに書き込むことができます．

5.2 CGのI/O制御命令を用いたサンプルプログラム

CGの持つ機能を，より具体的に理解していくために，これまで述べてきたCGのI/O制御命令を用いたサンプルプログラムを**リスト4-6**に示し，概説します．

このプログラムは，K-CG上のROMまたはRAMに定義・登録されているドットパターンを読み出し，それを拡大して画面に表示します．

リスト 4-6
CRTCのI/O制御命令を用いたサンプルプログラム

```
1 'save "CGREAD",a
2 '
3 '   Character Generator Font Read Sample Program V1.2
4 '   MS-DOS N88BASIC OK
100 SCREEN 3,0,0,1 : CLS 3
110 WIDTH 80,25 : CONSOLE ,,0,1
120 *LOOP
130 INPUT "漢字コード(&Hxxxx)=",A$
140 IF A$="" THEN CLS 2 : END
150 B$=LEFT$(A$,2) 'hex to decimal
160 IF B$="&h" OR B$="&H" THEN JC=VAL(A$) ELSE JC=VAL("&h"+A$)
170 JCH=JC ¥ 256 : JCL=JC MOD 256
180 IF JCH<&H21 OR &H7E<JCH THEN PRINT "&h2121 ～ &h7E7E" : GOTO *LOOP
190 IF JCL<&H21 OR &H7E<JCL THEN PRINT "&h2121 ～ &h7E7E" : GOTO *LOOP
200 CLS : LOCATE 0,0 : PRINT "&h";HEX$(JC);
210 LOCATE 0,1
220 DEF SEG=&HB800
230 OUT &H68,&HB                     'Dot map Select
240 OUT &HA1,JCL                     'Code low set
250 OUT &HA3,JCH -&H20               'Code high set
260 FOR I=0 TO 15                    'Font read
270   FOR J=1 TO 0 STEP -1
280     OUT &HA5,32*J+I
290     POKE I*80-J+100  ,INP(&HA9)
300     D=INP(&HA9)
310     FOR K=7 TO 0 STEP -1
320       W= (D AND 2^K)/2^K
330       IF W=1 THEN COLOR 7 ELSE  COLOR 0
340       PRINT "##";
350     NEXT K
360   NEXT J
370   PRINT
380 NEXT I
390 PRINT : COLOR 7
400 OUT &H68,&HA                     'Code map Select
410 GOTO *LOOP
420 END
```

6 CRT BIOS

6.1 CRT BIOSの手引き

これまで，第4章3，4，5でGDC，CRTC，CGのI/O制御命令について説明してきました．いずれのLSIも多くのI/O制御命令を持っていて，多機能であるのは確かなのですが，いざ，これら多数のI/O制御命令を複合して目的とする働きをさせようとすると，繁雑でとまどってしまいます．CRT BIOS*は，このデメリットを克服するために用意されているプログラムです．CRT BIOSは，いくつかのBIOSコマンドとして系統化されているので，ユーザも利用しやすくなっています．各BIOSコマンドは，I/O制御命令を複合化したものと考えることができます．そして，各BIOSコマンドには，BIOSコマンドコードが割り当てられています．

次に，CRT BIOSを利用する際の手続きについて説明します．

①レジスタAHに，CRT BIOSコマンドコードを設定する．

②必要があれば，他のレジスタあるいは所定のパラメータリスト領域に値を設定する（これらはBIOSコマンドにより異なる）．

③ソフトウェア割り込みを実行する

INT　18H（CRT BIOSの割り込みベクタコードは18H）

この手続きは，高級言語におけるサブルーチンコールの手続きによく似ています．つまり，BIOSにおいてレジスタやパラメータリストの値を設定するのは，サブルーチンコールする場合において引数を指定するのに対応しています．

なお，CRT BIOSコマンドは，次のように2つのグループに分けることができます．

CRT BIOSコマンド {テキスト画面表示制御用コマンド / グラフィック画面表示制御用コマンド}

個々のCRT BIOSについて，その機能とレジスタやパラメータリストの設定上の注意点をまとめて第4章6.2，6.3で述べます．各コマンドの説明は，下記

＊　BIOSについては，第3章3および5参照。

の5項目で構成されています．

項目名	解説
〔機能〕	コマンドの機能説明
〔コマンドコード〕	上記手続きの①に対応
〔入力〕	上記手続きの②に対応
〔割り込みコード〕	上記手続きの③に対応
〔出力〕	コマンド実行後に戻されるパラメータを列記している

6.2 テキスト画面制御用コマンド

CRT BIOSコマンドのうち，テキスト画面制御に関するものを取り上げて解説します．ここで解説するBIOSコマンドは，表4-11に示した12種類です．

表4-11 CRT BIOSコマンド(テキスト画面制御用)

No.	コマンド名	コマンド・コード
1	CRTモード設定／読出コマンド	0AH／0BH
2	テキスト画面表示ON／OFFコマンド	0CH／0DH
3	テキスト画面表示領域設定コマンド（画面分割なし）	0EH
4	テキスト画面表示領域設定コマンド（画面分割あり）	0FH
5	カーソル・ブリンクON／OF Fコマンド	10H
6	カーソル表示ON／OFFコマンド	11H／12H
7	カーソル位置設定コマンド	13H
8	フォント・パターン読出コマンド	14H
9	テキストVRAM初期化コマンド	16H
10	外字定義コマンド	1AH
11	K-CGアクセスモード設定コマンド	1BH

(1)CRTモード設定／読出コマンド

[機能]

T-GDC，CRTCのモード設定／読出を行う．

[割り込みコード]

INT　18H

[コマンドコード]（モード設定）

AH←0AH

[入力]

AL←0 0 0 0 b_3 b_2 b_1 b_0

[コマンドコード]（モード読出）

AH←0BH

[出力]

AL←b_7 0 0 0 b_3 b_2 b_1 b_0

		ビット＝0	ビット＝1
b_0	画面あたりの行数	25行	20行
b_1	行あたりの文字数	80文字	40文字
b_2	アトリビュート	垂線表示	簡易グラフ
b_3	K-CGのアクセスモード	コードアクセス	ドットアクセス
b_7	CRTの種別(読出のみ可)	標準	高解像度

(2)テキスト画面表示ON／OFFコマンド

[機能]

テキスト画面表示のON／OFFを指定する．

[割り込みコード]

INT　18H

[コマンドコード]

AH←0CH（表示ON）

AH←0DH（表示OFF）

(3)テキスト画面表示領域設定コマンド(画面分割なし)

[機能]

T-VRAM上の１つの領域をテキスト画面に割り付ける．

[割り込みコード]

INT　18H

[コマンドコード]

AH←0EH

[入力]

DX←T-VRAMの表示する領域の先頭アドレス

(CPUアドレスのオフセットで指定．セグメントはA000H固定)

(4)テキスト画面表示領域設定コマンド(画面分割あり)

[機能]

T-VRAM上の複数（最大4）の領域をテキスト画面に割り付ける．

[割り込みコード]

INT　18H

[コマンドコード]

AH←0FH

[入力]

BX←表示領域リストのセグメント・アドレス

CX←表示領域リストのオフセット・アドレス

DH←表示領域リストで最初に定義するエントリの表示領域番号

DL←表示領域リストで定義するエントリの個数（1～4）

＊表示領域リストのフォーマット

オフセット	内容	オフセット	内容
BX：CX＋00→	領域0の開始アドレス	BX：CX＋08→	領域2の開始アドレス
＋02→	領域0の行数	＋0A→	領域2の行数
＋04→	領域1の開始アドレス	＋0C→	領域3の開始アドレス
＋06→	領域1の行数	＋0E→	領域3の行数

・表示領域リストはDLで指定した領域数分だけ用意すればよい．
　DL＝4の時に最大となり，16バイト必要．

・開始アドレスはGDCアドレスで指定する（図4-7，4-8参照）．
　つまり，画面左上隅が0000H，右上隅が004FH．

・行数は，20行モードでは1～20，25行モードでは1～25．

⑸カーソルのブリンク状態のON・OFFコマンド

［機能］

カーソルのブリンク状態のON・OFFを設定する.

［割り込みコード］

INT　18H

［コマンドコード］

AH←10H

［入力］

AL ←01H：OFF
　　　00H：ON

⑹カーソル表示のON・OFFコマンド

［機能］

カーソル表示のON・OFFを設定する.

［割り込みコード］

INT　18H

［コマンドコード］

AH ←11H：ON
　　　12H：OFF

⑺カーソル位置設定コマンド

［機能］

カーソルの表示位置をT-VRAMに割り当てられたCPUアドレスで設定する.

［割り込みコード］

INT　18H

［コマンドコード］

AH←13H

［入力］

DX←表示位置（T-VRAMのCPUアドレスのオフセットで指定，セグメントはA000H固定）

⑻フォントパターン読み出しコマンド

[機能]

ANK文字や日本字のコードを指定し,指定した文字のフォントパターンをフォントパターンバッファへ出力する.

[割り込みコード]

INT　18H

[コマンドコード]

AH←14H

[入力]

BX←フォントパターンバッファの先頭アドレス*（セグメントアドレス）

CX←フォントパターンバッファの先頭アドレス（オフセットアドレス）

DX←文字コード**

⑼T-VRAMの初期化コマンド

[機能]

T-VRAMの全領域をユーザが指定する文字コードで埋める.

[割り込みコード]

INT　18H

[コマンドコード]

AH←16H

[入力]

DH←アトリビュート領域を埋めるデータ

DL←文字コード領域を埋める文字コード

---

*　フォントパターンバッファへ出力されたデータの形式を以下に示す.

内部作業域（2バイト）	データ

↑
BX：CX

データのサイズは，文字の種類に依存する.

ANK文字，日本字（¼角）：8バイト
日本字（半角）：16バイト
日本字（全角）：32バイト

**　文字コードの設定法

ANK文字の場合 { DL←ASCIIコード / DH←80H }
日本字の場合　DX←JISコード

(10)ユーザ文字定義コマンド

［機能］

ユーザ独自の文字・記号のフォントパターンをK-CG上のRAMに格納し，そのパターンに対して文字コードを登録する．

［割り込みコード］

INT　18H

［コマンドコード］

AH←1AH

［入力］

BX←フォントパターンバッファ*の先頭アドレス（セグメントアドレス）

CX←フォントパターンバッファの先頭アドレス（オフセットアドレス）

DX←登録コード　VF/VM/UV/VX/UX(7620H～767FH, 7720H～777FH)
　　　　　　　　E/F/M/U(7620H～765FH)

(11)K-CGアクセスモード設定コマンド

［機能］

K-CGのアクセスモードをドットアクセスモード，またはコードアクセスモードに設定する．

［割り込みコード］

INT　18H

［コマンドコード］

AH←1BH

［入力］

AL ←00H：コードアクセスモード
　　01H：ドットアクセスモード

注）グラフィック画面へは，いずれのモードの場合でも出力できる．
　テキスト画面へは，コードアクセスモードの場合のみ出力できる．

---

* ユーザは，このコマンド実行に先立ち，フォントパターンバッファに登録したいフォントパターンを格納しておかなければならない．フォントパターンへのデータ格納形式を以下に示す．

内部制御域（2バイト）	データ（32バイト）

6.3 グラフィック画面制御用コマンド

CRT BIOSコマンドのうち，グラフィック画面制御に関するものを取り上げて解説します．ここで解説するBIOSコマンドは，**表4-12**で示した10種類です．

なお，CRT BIOS（グラフィック制御用）を使用する際の注意点を以下にまとめて示します．

①スタックエリアの確保

ユーザ自身がスタックエリアとして，30バイト以上を確保する必要があります．スタックエリアの設定はSS（スタックセグメントレジスタ）とSP（スタックポインタ）で行います．

②CPUステータスフラグの設定

ステータスフラグのビット設定	解説
IF =	割り込み受付可能状態
TF =	シングルステップモードクリア状態

③UCW*（制御情報域）の確保

描画情報の受け渡しや保存のために約800バイトのメモリ領域を確保しておく必要があります．UCWは，複数のフィールド（制御パラメータ）で構成されています．

表4-12　CRT BIOSコマンド（グラフィック画面制御用）

No.	コマンド名	BIOSコマンドコード
1	グラフィック画面表示のON，OFFコマンド	40H/41H
2	表示領域設定コマンド	42H
3	パレットレジスタ設定コマンド	43H
4	ボーダカラー設定コマンド	44H
5	G-VRAMへのドット書き込みコマンド	45H
6	G-VRAMからのドット読み出しコマンド	46H
7	直線・矩形描画コマンド	47H
8	円弧描画コマンド	48H
9	グラフィック文字の書き込みコマンド	49H
10	高速描画設定コマンド	4AH

＊　UCW＝Unit Control Work

(1)グラフィック画面表示のON・OFFコマンド

[機能]

グラフィック画面表示のON・OFFを設定する.

[割り込みコード]

INT 18H

[コマンドコード]

AH ←40H：ON

41H：OFF

(2)表示領域設定コマンド

[機能]

グラフィック画面モードの選択と，カラーかモノクロかの選択を行う.

[割り込みコード]

INT 18H

[コマンドコード]

AH←42H

[入力]

CH← $b_7 b_6 b_5 b_4 0\ 0\ 0\ 0$ 　設定データ

ビット番号	解　　説	ビット値＝0	ビット値＝1
b_4	G-VRAM(1)，(2)の選択	G-VRAM(1)	G-VRAM(2)
b_5	モノクロかカラーかの選択	カラー	モノクロ
$b_7 b_6$＝	表示するG-VRAMの領域を指定 01：前半16Kバイトを表示（UPPERモード）(640×200) 10：後半16Kバイトを表示（LOWERモード）(640×200) 11：32Kバイト全体を表示（ALLモード）(640×400)		

⑶パレットレジスタ設定コマンド

［機能］

カラーモード時には，パレットレジスタにカラーコードを設定する．

モノクロモード時には，表示画面の選択・合成*の仕方を設定する．

［割り込みコード］

INT　18H

［コマンドコード］

AH←43H

［入力］

DS←UCW**の先頭アドレスのセグメントアドレス

BX←UCWの先頭アドレスのオフセットアドレス

---

*　図4－10参照

**　ユーザは，コマンドの実行に先立ち，ＵＣＷ内のパラメータＧＢＣＰＣを下記のとおり設定しておかなければならない．ＵＣＷとは，ＣＲＴ　ＢＩＯＳコマンドに付随するパラメータなどを格納納するメモリ領域のことである．

相対アドレス	制御パラメータ	サイズ(バイト)	機　　能
4H	GBCPC	1	パレット#6，#7のカラーコードを設定***
5H		1	パレット#4，#5の
6H		1	パレット#2，#3の
7H		1	パレット#0，#1の

注)相対アドレスとは，レジスタDS,BXで設定したUCWの先頭アドレスを基準(0H)にしたアドレスである．

***　ＧＰＣＰＣのデータとパレットレジスタとの対応関係を以下に示す．

第1バイト		第2バイト		第3バイト		第4バイト	
MSB → LSB		MSB → LSB		MSB → LSB		MSB → LSB	
パレット#6	パレット#7	パレット#4	パレット#5	パレット#2	パレット#3	パレット#0	パレット#1
0 G R B	0 G R B	0 G R B	0 G R B	0 G R B	0 G R B	0 G R B	0 G R B

注）ＧＲＢのビット値の組み合わせて，各パレットの色を指定する．

⑷ボーダカラー設定コマンド

[機能]

CRTのボーダカラーを設定する（ボーダカラーレジスタにカラーコードを設定する）.

[割り込みコード]

INT　18H

[コマンドコード]

AH←44H

[入力]

DS←UCW*の先頭アドレスのセグメントアドレス

BX←UCWの先頭アドレスのオフセットアドレス

---

* ユーザは，このコマンドの実行に先立ち，ＵＣＷ内のパラメータをＧＢＢＣＣを下記のとおり設定しておかなければならない.

相対アドレス	制御パラメータ	サイズ(バイト)	機能
1H	GBBCC	1	ボーダカラーレジスタに設定するカラーコードを格納

注）相対アドレスとは，ＵＣＷの先頭アドレスを基準にしたアドレスである.

⑸G-VRAMへのドット書き込みコマンド

[機能]

G-VRAMに対して，ドット単位の書き込みを行う．書き込みの対象となる描画画面の選択も行う．

[割り込みコード]

INT 18H

[コマンドコード]

AH←45H

[入力]

CH←描画画面を選択する値　　$b_7 b_6 b_5 b_4 0000$

$b_5 b_4$	G-VRAM上のプレーンの指定．図4-2参照．
=	00　Bプレーンのみ
=	01　Rプレーンのみ
=	10　Gプレーンのみ
=	11　B，G，Rプレーンすべて
b_6	各プレーンの描画範囲．
=	0　プレーン全体(ALL)または2等分したプレーンの前半(LOWER)
=	1　2等分したプレーンの後半 (UPPER)
b_7	画面解像度
=	0　600×200ドット
=	1．　600×400ドット

ES←描画パターンバッファのセグメントアドレス
DS←UCWのセグメントアドレス*
BX←UCWのセグメントアドレス

＊ ユーザは，このコマンドの実行に先立ち，ＵＣＷ内のパラメータを下記のとおり設定しておかなければならない．

相対アドレス	制御パラメータ	サイズ(バイト)	機能
0H	GBON-PTN	1	３プレーン同時描画時のオペレーション設定
2H	GBDOTU	1	単一プレーン描画時のオペレーション設定
8H	GBSX1	2	描画開始点のＸ座標 } オリジナルスクリーン座標
AH	GBSY1	2	描画開始点のＹ座標 } オリジナルスクリーン座標
CH	GBLNG1	2	書き込み長さ（ドット単位）
EH	GBWDPA	2	描画パターンバッファの先頭オフセットアドレス

GBON-PTN＝ $00000 b_2 b_1 b_0$

b_0，b_1，b_2＝０：B，G，Rプレーンをクリア
b_0，b_1，b_2＝１：B，G，Rプレーンをセット

GBDOTU＝00H：REPLACE　　＝02H：CLEAR　} ドット修正モード参照
　　　　01H：COMPLEMENT　＝03H：SET

注）相対アドレスとは，ＵＣＷの先頭アドレスを基準にしたアドレスである．

(6)G-VRAMからのドット読み出しコマンド

[機能]

G-VRAM上の指定した描画画面から，ドット単位の読み出しを行い，それを指定した読み込みバッファに格納する．

[割り込みコード]

INT　18H

[コマンドコード]

AH←46H

[入力]

CH←描画画面を選択する値．(5)項参照

DS←UCWのセグメントアドレス*

BX←UCWのオフセットアドレス

ES←読み込みバッファのセグメントアドレス

---

* ユーザは，このコマンドの実行に先立ち，UCW内のパラメータを下記のとおり設定しておかなければならない．

相対アドレス	制御パラメータ	サイズ(バイト)	機能
8H	GBSX1	2	画面上の読み込み開始点のX座標
AH	GBSY1	2	画面上の読み込み開始点のY座標
CH	GBLNG1	2	読み込む長さ（単位：ドット）
10H	GBRBUF1	2	読み込みバッファ1の先頭オフセットアドレス(Bプレーン用)
12H	GBRBUF2	2	読み込みバッファ2の先頭オフセットアドレス(Rプレーン用)
14H	GBRBUF3	2	読み込みバッファ3の先頭オフセットアドレス(Gプレーン用)

注）相対アドレスとは，UCWの先頭アドレスを基準にしたアドレスである．

(7)直線・矩形描画コマンド

[機能]

G-VRAM上の指定した描画画面に，直線（破線も含む）や矩形を書き込む.

[割り込みコード]

INT　18H

[コマンドコード]

AH←47H

[入力]

CH←描画画面を選択する値　(5)項参照

DS←UCWのセグメントアドレス*

BX←UCWのオフセットアドレス

---

* ユーザは，このコマンドの実行に先立ち，UCW内のパラメータを下記のとおりに設定しておかなければならない.

相対アドレス	制御パラメータ	サイズ(バイト)	機能
0H	GBON-PTN	1	3プレーン同時書き込み時のオペレーション設定 (5)項参照
2H	GBDOTU	1	単一プレーン書き込み時のオペレーション設定 (5)項参照
3H	GBDSP	1	描画方向
8H	GBSX1	2	画面上の書き込み開始点のX座標
AH	GBSY1	2	画面上の書き込み開始点のY座標
16H	GBSX2	2	画面上の書き込み終了点のX座標
18H	GBSY2	2	画面上の書き込み終了点のY座標
20H	GBLPTN	2	線種パターン
28H	GBDTYP	1	直線・矩形の選択
		=	01H　直線
		=	02H　矩形

注）相対アドレスとは，UCWの先頭アドレスを基準にしたアドレスである.

第4章 グラフィックス

描画方向

描画方向制御 パラメータ値	直線	矩形	円弧
0			
1			
2			
3			
4			
5			
6			
7			

注）・開始点　GBSX 1 , GBSY 1　∘終了点　GBSX 2 , GBSY 2

⑻円弧描画コマンド

[機能]

G-VRAM上の指定した描画画面上に円弧を描画する。

[割り込みコード]

INT　18H

[コマンドコード]

AH←48H

[入力]

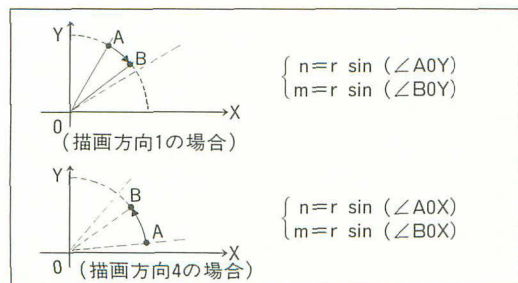
CH←描画画面を設定する値．⑸項参照

DS←UCWのセグメントアドレス*

BX←UCWのオフセットアドレス

---

* ユーザは，このコマンドの実行に先立ち，ＵＣＷ内のパラメータを下記のとおり設定しておかなければならない．

相対アドレス	制御パラメータ	サイズ(バイト)	機能
0H	GBON-PTN	1	3プレーン同時書き込み時のオペレーション設定 ⑸項参照
2H	GBDOTU	1	単一プレーン書き込み時のオペレーション設定 ⑸項参照
3H	GBDSP	1	描画方向 ⑺項参照
8H	GBSX1	2	開始点のＸ座標
AH	GBSY1	2	開始点のＹ座標
CH	GBLNG1	2	描画総ドット数　m 注)
1AH	GBMDOT	2	マスキングドット数　n 注)
1CH	GBCIR	2	半径　r 注)
20H	GBLPTN	2	線種パターン
28H	GBDTYP	1	04H　円弧を指定

注）中心点を0，半径ｒ，描画開始点をＡ，終了点をＢとする。

⑼グラフィック文字の書き込みコマンド

[機能]

G-VRAM上にグラフィック文字を書き込む.グラフィック文字を8×8ドット以下のサイズの基本パターンとして定義する.指定領域の基本パターンを繰り返しながら書き込みを行う.

[割り込みコード]

INT 18H

[コマンドコード]

AH←49H

[入力]

CH←描画画面を選択する値.⑸項参照.

DS←UCWのセグメントアドレス

BX←UCWのオフセットアドレス

---

* ユーザは,このコマンドの実行に先立ち,UCW内のパラメータを下記のとおり設定しておかなければならない.

相対アドレス	制御パラメータ	サイズ(バイト)	機能
0H	GBON-PTN	1	⑸項参照
2H	GBDOTU	1	
3H	GBDSP	1	描画方向　⑺項参照
8H	GBSX1	2	開始点のX座標
AH	GBSY1	2	開始点のY座標
CH	GBLNG1	2	描画領域のX方向ドット数
1EH	GBLNG2	2	描画領域のY方向ドット数
20H	GBDOTI	8	8×8ドットの基本パターンを格納

⑽高速描画設定コマンド

［機能］

2つの描画モード（フラッシュ描画/フラッシュレス描画*）のうち，フラッシュ描画モードを選択すれば，描画速度を5倍に速めることができる．

［割り込みコード］

INT 18H

［コマンドコード］

AH←4AH

［入力］

CH←描画モードの設定 06H：フラッシュ描画 16H：フラッシュレス描画

≡6.4≡ CRT BIOSを用いたサンプルプログラム

ＣＲＴ ＢＩＯＳを用いたサンプルプログラムをいくつか紹介して，簡単に解説します．

⑴サンプルプログラムA

K-CG内の漢字ROMに格納されている漢字のフォントパターンを画面に表示するプログラムを**リスト4-7**に示します．

BASICプログラムをRUNすると，漢字コードを要求してきますから，それに応じて入力すればそのコードに対する漢字のフォントパターンが画面に表示されます．

* フラッシュレス描画では，ＣＲＴへの表示動作時間以外のタイミングで描画を行うので，表示画面は安定している．
  フラッシュ描画では，表示動作中にも描画を行うので，描画速度は向上するが，画面にフラッシュが発生する（フラッシュが発生するのはE/F/Mのみ）．

リスト 4-7

CRT BIOSサンプルプログラムA(BASIC)

```
1 'save "FONT",a
2 '
3 '       Font pattern read V1.2
4 '       MS-DOS N88BASIC OK
100 WIDTH 80,25
110 SCREEN 3,0,0,1 : CLS 3
120 DEF SEG=0 : A=(PEEK(&H501) AND 7)+1 '機械語セグメント算出
130 CSEG=&H2000*A-&H100
140 DEF SEG=CSEG
150 FOR A=0 TO &HF
160   READ A$ : POKE A,VAL("&H"+A$)      '機械語書き込み
170 NEXT
180 *LOOP
190 LOCATE 0,0 : PRINT "1/4角[0000～00FF], 半角[2921～297B,297D,2A20～2A5F], ";
200 LOCATE 0,1 : PRINT "全 角[2121～7E7E], ANK [8000～80FF]"
210 LOCATE 0,2 : PRINT SPACE$(79);
220 LOCATE 2,2 : INPUT "文字コード=",A$
230 IF A$="" THEN CLS 2 : END
240 B$=LEFT$(A$,2)
250 IF B$="&h" OR B$="&H" THEN JC=VAL(A$) ELSE JC=VAL("&h"+A$)
260 IF JC<0 THEN JC=JC+65536!
270 JCH=INT(JC / 256) : JCL=JC - JCH*256
280 DEF SEG=CSEG
290 POKE &H42,JCL
300 POKE &H43,JCH
310 A=0 : CALL A
320 A=PEEK(&H20) : B=PEEK(&H21)
330 IF A=2 AND B=2                THEN A$="全 角" : L=31 : F=1 : GOTO *FONTWR
340 IF A=1 AND B=1                THEN A$="1/4角" : L=7  : F=0 : GOTO *FONTWR
350 IF A=2 AND B=1 AND JCH>&H7F THEN A$="ANK  " : L=15 : F=0 : GOTO *FONTWR
360 IF A=2 AND B=1 AND JCH<&H7F THEN A$="半 角" : L=15 : F=0 : GOTO *FONTWR
370 *FONTWR      'フォント出力
380 LINE(127,66)-(144,84),0,BF
390 LOCATE 4,4 : PRINT A$;
400 FOR I=0 TO L
410   FOR J=0 TO F
420     DEF SEG=CSEG   : PTN=PEEK(&H22+I+J)
430     DEF SEG=&HB800 : POKE I*&H50/(F+1)+J+&H1500,PTN
440   NEXT J
450   I=I+F
460 NEXT I
470 GOTO *LOOP
480 END
490 DATA 0E             'PUSH   CS
500 DATA 1F             'POP    DS
510 DATA B4,14          'MOV    AH,14
520 DATA 8C,DB          'MOV    BX,DS
530 DATA B9,20,00       'MOV    CX,20
540 DATA 8B,16,42,00    'MOV    DX,[42]
550 DATA CD,18          'INT    18
560 DATA CF             'IRET
```

(2)サンプルプログラムB

CRT BIOSを用いたサンプルプログラムBのリストを**リスト4-8**に示します．これは，BASICインタープリタのグラフィックコマンド

LINE (100, 100) - (400, 200), 5, B

の機能を，CRT BIOSを用いて表現したものです．

リスト 4-8
CRT BIOSサンプルプログラムB

```
                              ;
                              ;  GBIOS-SAMPLE
                              ;    LINE(100,100)-(400,200),5,B
                              ;
                              ;
                              cseg
                              org 0h
                              ;
                              ;  INITIALIZE
                              ;
0000 8CC8                     MOV AX,CS
0002 8ED8                     MOV DS,AX
0004 8ED0                     MOV SS,AX
0006 8D061201                 LEA AX,STACK_BOT
000A 8BE0                     MOV SP,AX
000C FB                       STI
                              ;
                              ;  START DISPLAY COMMAND
                              ;
000D B440                     MOV AH,40H ───── 画面表示
000F CD18                     INT 18H
                              ;
                              ;  SET DISPLAY AREA
                              ;
0011 B442                     MOV AH,42H ───── 表示領域設定
0013 B5C0                     MOV CH,0C0H     ;ALL
0015 CD18                     INT 18H
                              ;
                              ;  MAIN
                              ;
0017 B447                     MOV AH,47H ───── 直線画面
0019 B5B0                     MOV CH,0B0H
001B BB2100                   MOV BX,OFFSET DATA
001E CD18                     INT 18H
0020 F4                       HLT
                              ;
                              DATA:
0021 05                       GRON_PTN      DB 5              ;COLOR
0022 00                       GRBCC         DB 0
0023 03                       GRDOTU        DB 3              ;PSET
0024 00                       GRDSP         DB 0
0025 00000000                 GRCPC         DB 0,0,0,0
0029 6400                     GRSX1         DW 100            ;SI-TEN
002B 6400                     GRSY1         DW 100
002D 0000                     GRLNG1        DW 0
002F 0000                     GRWDPA        DW 0
0031 000000000000             GRRBUF        DW 0,0,0
0037 9001                     GRSX2         DW 400            ;SHU-TEN
0039 C800                     GRSY2         DW 200
003B 0000                     GRMDOT        DW 0
003D 0000                     GRCIR         DW 0
003F 0000                     GRLNG2        DW 0
0041 FFFF                     GRLPTN        DW 0FFFFH
                                            ORG (OFFSET $)-2
0041 000000000000             GRDOTI        DW 0,0,0,0
     0000
0049 02                       GRDTYP        DB 2
                              ;
                              ;  STACK AREA
                              ;
004A                          STACK_TOP     RW 100
0112                          STACK_BOT     RW 1
                              ;
                              END
```

(3)サンプルプログラムC

CRT BIOSを用いたサンプルプログラムCのリストを**リスト4-9**に示します．これは，BASICインタープリタのグラフィックコマンドであるCIRCLE命令をCRT BIOSを用いて表現したものです．

CRT BIOSの円弧描画コマンドでは一度に1/8円弧しか描けないので，パラメータを変更しながら8回実行して円を描いています．円の中心＝（200，200)，半径＝100，色＝5，線種＝F0F0Hに設定しています．

リスト 4-9
CRT BIOSサンプルプログラムC

```
                                  ;  GBIOS-SAMPLE
                                  ;    CIRCLE(200,200),100,5,&HF0F0
                                  ;
                                  CSEG
                                  ORG 0H
                                  ;
                                  ;  INITIALIZE
                                  ;
0000 8CC8                         MOV AX,CS
0002 8ED8                         MOV DS,AX
0004 8ED0                         MOV SS,AX
0006 8D06C001                     LEA AX,STACK_BOT
000A 8BE0                         MOV SP,AX
000C FB                           STI
                                  ;
                                  ;  START DISPLAY COMMAND
                                  ;
000D B440                         MOV AH,40H―――画面表示
000F CD18                         INT 18H
                                  ;
                                  ;  SET DISPLAY AREA
                                  ;
0011 B442                         MOV AH,42H――― 表示領域設定
0013 B5C0                         MOV CH,0C0H      ;ALL
0015 CD18                         INT 18H
                                  ;
                                  ;  MAIN
                                  ;
0017 BBCF00                       MOV BX,OFFSET DATA
001A 2E8B16CB00                   MOV DX,SX1
001F 2E0316EB00                   ADD DX,GBCIR
0024 2E8916D700                   MOV GBSX1,DX
0029 2E8B16CD00                   MOV DX,SY1
002E 2E8916D900                   MOV GBSY1,DX
0033 2EC606D20007                 MOV GBDSP,7
0039 B448                         MOV AH,48H―――円弧描画
003B B5B0                         MOV CH,0B0H
003D CD18                         INT 18H
                                  ;
003F 2EC606D20004                 MOV GBDSP,4
0045 B448                         MOV AH,48H―――円弧描画
0047 B5B0                         MOV CH,0B0H
0049 CD18                         INT 18H
                                  ;
004B 2E8B16CB00                   MOV DX,SX1
0050 2E2B16EB00                   SUB DX,GBCIR
0055 2E8916D700                   MOV GBSX1,DX
005A 2EC606D20003                 MOV GBDSP,3
0060 B448                         MOV AH,48H―――円弧描画
```

```
0062 B5B0                MOV CH,0B0H
0064 CD18                INT 18H
                         ;
0066 2EC606D20000        MOV GBDSP,0
006C B448                MOV AH,48H————————円弧描画
006E B5B0                MOV CH,0B0H
0070 CD18                INT 18H
                         ;
0072 2E8B16CB00          MOV DX,SX1
0077 2E8916D700          MOV GBSX1,DX
007C 2E8B16CD00          MOV DX,SY1
0081 2E2B16EB00          SUB DX,GBCIR
0086 2E8916D900          MOV GBSY1,DX
008B 2EC606D20006        MOV GBDSP,6
0091 B448                MOV AH,48H————————円弧描画
0093 B5B0                MOV CH,0B0H
0095 CD18                INT 18H
                         ;
0097 2EC606D20001        MOV GBDSP,1
009D B448                MOV AH,48H ———————円弧描画
009F B5B0                MOV CH,0B0H
00A1 CD18                INT 18H
                         ;
00A3 2E8B16CD00          MOV DX,SY1
00A8 2E0316EB00          ADD DX,GBCIR
00AD 2E8916D900          MOV GBSY1,DX
00B2 2EC606D20002        MOV GBDSP,2
00B8 B448                MOV AH,48H ———————円弧描画
00BA B5B0                MOV CH,0B0H
00BC CD18                INT 18H
                         ;
00BE 2EC606D20005        MOV GBDSP,5
00C4 B448                MOV AH,48H ———————円弧描画
00C6 B5B0                MOV CH,0B0H
00C8 CD18                INT 18H
                         ;
00CA F4                  HLT
                         ;
                         SUBDATA:
00CB C800                SX1             DW 200          ;CHUSIN
00CD C800                SY1             DW 200
                         ;
                         DATA:
00CF 05                  GBON_PTN        DB 5            ;COLOR
00D0 00                  GBBCC           DB 0
00D1 03                  GBDOTU          DB 3            ;PSET
00D2 00                  GBDSP           DB 0
00D3 00000000            GBCPC           DB 0,0,0,0
00D7 0000                GBSX1           DW 0
00D9 0000                GBSY1           DW 0
00DB 4700                GBLNG1          DW 71
00DD 0000                GBWDPA          DW 0
00DF 000000000000        GBRBUF          DW 0,0,0
00E5 0000                GBSX2           DW 0
00E7 0000                GBSY2           DW 0

00E9 0000                GBMDOT          DW 0
00EB 6400                GBCIR           DW 100          ;HANKEI
00ED 0000                GBLNG2          DW 0
00EF F0F0                GBLPTN          DW 0F0F0H
                                         ORG (OFFSET $)-2
00EF 000000000000        GBDOTI          DB 0,0,0,0,0,0,0,0
     0000
00F7 04                  GBDTYP          DB 4
                         ;
                         ;  STACK AREA
                         ;
00F8                     STACK_TOP       RW 100
01C0                     STACK_BOT       RW 1
                         ;
                         END
```

7 グラフィックLIO

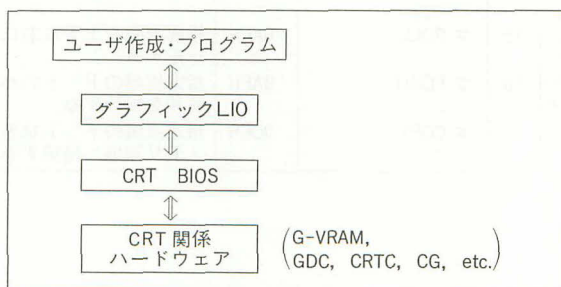
7.1 グラフィックLIOの概要

PC-98では，多彩な機能を実現するためにグラフィック表示のための処理を行う専用LSI (GDC，CRTC，CG, etc) を使用しています．これらLSIの持つ機能については，すでに述べた通りです．そして，これらLSIがハードウェアとして持つ潜在的な能力を，より簡単な操作で最大限に引き出すための基本ソフトウェアとして，CRT　BIOSが用意されていることも第4章6で述べました．CRT BIOSが画面表示に関するハードウェアを直接制御しているわけです．

ここでは，グラフィックLIO*について説明しますが，これはCRT BIOSの上位に位置づけられるソフトウェアであり，CRT BIOSをさらに複合化して，17種類のコマンドに系統化したものです．各グラフィックLIOコマンドには，**表4-13**に示すようにコマンドコードが割り当てられています**．

PC-98システムのソフトウェア上におけるグラフィックLIOの階層的位置づけを示します（**図4-17**参照）．

図4-17
システムにおけるグラフィックLIOの位置づけ

---

*　LIO＝Logical Input Output

**　ここで言うコマンドコードとは，すなわち割り込みベクタコードに相当する（表3-1参照）

表4-13　グラフィックLIOコマンドとコマンドコード

No.	グラフィックLIO コマンド名	ベクタ コード	説明	対応する BASICコマンド
1	＃INIT	0A0H	グラフィックLIOの初期化を行う	
2	＃ SCREEN	0A1H	グラフィック画面のモード設定を行う	SCREEN
3	＃ VIEW	0A2H	描画領域を指定する	VIEW
4	＃ COLOR 1	0A3H	背景色を指定する	COLOR
5	＃ COLOR 2	0A4H	パレットレジスタの設定をする	COLOR(　,　)
6	＃ CLS	0A5H	描画領域を背景色で塗りつぶす	CLS 2
7	＃ PSET	0A6H	点を打つ	PSET/PRESET
8	＃ LINE	0A7H	直線・矩形を描く	LINE
9	＃ CIRCLE	0A8H	円・楕円を描く	CIRCLE
10	＃ PAINT 1	0A9H	指定領域を指定色で塗りつぶす	PAINT
11	＃ PAINT 2	0AAH	指定領域を指定タイルパターンで塗りつぶす	PAINT
12	＃ GET	0ABH	指定領域の描画情報を指定メモリ領域へ格納する	GET
13	＃ PUT 1	0ACH	＃GETの逆操作	PUT
14	＃ PUT 2	0ADH	日本字を指定領域に描く	PUT
15	＃ ROLL	0AEH	描画画面を上下左右にスクロールする	ROLL
16	＃ POINT	0AFH	指定座標のドットのパレット番号を検知する	？＝POINT(　,　)
17	＃ COPY	0CEH	指定領域のドット状態を指定メモリ領域へ格納する	

7.2 グラフィックLIOの使用法

次に，グラフィックLIOを使用する際の準備と使用法について説明します．

(1)割り込みベクタテーブルの設定

グラフィックLIOの各コマンドは，ソフトウエア割り込みによって呼び出します．**表4-13**でコマンドコードと呼んでいるコードは割り込みベクタコードに相当するものです．

ソフトウエア割り込みについては，すでに第３章３で述べましたが，**表3-1**から明らかなように，ベクタコードA0H～AFH，CEHがグラフィックLIOに割り当てられています．

N_{88}-BASICで使用する場合には，グラフィックLIOの各ベクタコードに対応する割り込み先のアドレス（セグメントアドレスとオフセットアドレス）が割り込みベクタテーブルに自動的に記入されるようになっています．しかし，他のOS（CP/M，MS-DOS etc）でグラフィックLIOを使用する場合には，ユーザ自身がベクタテーブルの内容を準備しなければなりません．

グラフィックLIOはROM上に存在し，先頭のCPUアドレス（オフセットアドレス）が**表4-14**に示すようにテーブルとしてまとめられています．

N_{88}-BASIC以外でグラフィックLIOを使用する場合には，ユーザは**表4-14**に基づいてROMから各LIOコマンドのオフセットアドレスを読み出し，それを割り込みベクタテーブルに転記する必要があります．

なお，グラフィックLIOにおける割り込みベクタ設定のサンプルプログラムを**リスト4-10**に示します．ただし，このプログラムのほかに，長時間にわたる描画処理の中断を可能にする割り込みベクタC5Hをユーザが設定する必要があります．また，グラフィックLIOで使用するレジスタおよびハード/ソフトの状態の保存が必要です．例えば，このルーチンは実処理を伴わないIRETのみのルーチンでも可能です．

表4-14　グラフィックLIOの割り込みベクタテーブル

物理アドレス	第1バイト	第2バイト	第3バイト	第4バイト	備　考
F9900H	11H	×	×	×	11Hはエントリ数
+04H	A0H	00H	#INTのオフセットアドレス		
+08H	A1H	00H	#SCREENのオフセットアドレス		
+0CH	A2H	00H	#VIEWのオフセットアドレス		
+10H	A3H	00H	#COLOR1のオフセットアドレス		
+14H	A4H	00H	#COLOR2のオフセットアドレス		
+18H	A5H	00H	#CLSのオフセットアドレス		
+1CH	A6H	00H	#PSETのオフセットアドレス		
+20H	A7H	00H	#LINEのオフセットアドレス		
+24H	A8H	00H	#CIRCLEのオフセットアドレス		
+28H	A9H	00H	#PAINT1のオフセットアドレス		
+2CH	AAH	00H	#PAINT2のオフセットアドレス		
+30H	ABH	00H	#GETのオフセットアドレス		
+34H	ACH	00H	#PUT1のオフセットアドレス		
+38H	ADH	00H	#PUT2のオフセットアドレス		
+3CH	AEH	00H	#ROLLのオフセットアドレス		
+40H	AFH	00H	#POINTのオフセットアドレス		
+44H	CEH	00H	#COPYのオフセットアドレス		

リフト4-10

グラフィックLIOの割り込みベクタ設定
サンプルプログラム

```
0100 31C0          XOR    AX,AX
0102 8EC0          MOV    ES,AX
0104 B890F9        MOV    AX,F990
0107 8ED8          MOV    DS,AX
0109 8B0E0000      MOV    CX,[0000]
010D FC            CLD
010E BE0400        MOV    SI,0004
0111 8B3C          MOV    DI,[SI]
0113 01FF          ADD    DI,DI
0115 01FF          ADD    DI,DI
0117 46            INC    SI
0118 46            INC    SI
0119 A5            MOVSW
011A 26            ES:
011B 8905          MOV    [DI],AX
011D E2F2          LOOP   0111
011F C3            RET
```

⑵グラフィックLIOの初期設定

まず，グラフィックLIOを使用するに際して，専用のワークエリアとスタックエリアをデータセグメント上に作成しておく必要があります．それぞれの所要メモリサイズを以下に示します．

メモリ・領域名		メモリサイズ
ワークエリア	＃COPYコマンド使用時	1400Hバイト（5120バイト）
	上記以外のコマンドの時	1200Hバイト（4608バイト）
スタックエリア		80Hバイト（128バイト）

なお，ワークエリアはデータセグメント上に，オフセットアドレス0000Hから作成されます．ワークエリアのメモリマップを**図4-18**に示します．

ユーザは，データセグメントを使用する場合，このワークエリアに不用意にアクセスして，データを破壊しないように気を付ける必要があります．

なお，ワークエリアの先頭部分が未使用状態なので，この領域をスタックエリアとして使用することにします．

それぞれのグラフィックLIOコマンドには，固有のパラメータがあります．ユーザは，このパラメータのリストをデータセグメント上に設定しておく必要があります．パラメータエリアの先頭アドレスをレジスタBXで指定しますが，データセグメントにおけるオフセットアドレスで表現したものを用います．この場合，パラメータエリアがワークエリアを侵害しないように気を付けて下さい．

図4-18　グラフィックLIOワークエリアのメモリマップ

0000H	0620H	06ADH	1000H	1200H	1380H
未使用	共通作業領域	未使用	個別作業領域	未使用	＃COPY作業領域

1400H

（注）アドレスは，データセグメントにおけるオフセットアドレスを示している．
＊　０Ａ08Ｈ～56Ｂ：共通作業領域拡張部(Ｕ/ＶＦ/ＶＭ/ＵＶ)

ワークエリアのメモリマップから明らかなように，BXには，例えば1500Hを設定しておけば問題ありません．

ワークエリア，スタックエリア，パラメータエリアを設定するプログラムを具体的に示します（**リスト4-11**参照）．

ユーザは，このパラメータエリアの指定されたアドレスに必要な値を設定しておきます．以上の準備ができた段階で，下記のソフトウェア割り込みを実行すれば目的とするグラフィックLIOコマンドが実行されます．

INT　m（mはグラフィックLIOコマンドコード）

リスト 4-11

ワークエリア，スタックエリア，パラメータエリアの設定プログラム

```
MOV AX,60H   }データセグメントのセグメントアドレスを設定する
MOV DS,AX    }(ワークエリアはデータセグメントの先頭から作成される)

MOV SS,AX    }
MOV AX,200H  }ワークエリアの未使用領域にスタックエリアを設定する
MOV SP,AX    }

MOV BX,1500H ——パラメータエリアの先頭アドレスを設定する
```

※図2-5のメモリマップを参照

7.3 グラフィックLIOコマンドの解説

グラフィックLIOコマンドは，表4-13に示したように17種類あります．ここでは，個々のコマンドについて解説します．

各コマンドの説明文中で用いている略称などの説明を以下に示します．

記号，略称	説明
[コード]	コマンドコード(ソフトウェア割り込み実行文の形で示している)
[レジスタ]	設定すべきレジスタを列記している． なお，リスト4-6のプログラムは，下記表現と等価である． DS ← 60H SS ← 60H SP ← 200H BX ← 1500H
[パラメータ]	設定すべきパラメータを列記している． なお，レジスタBXの値が，パラメータエリアの先頭アドレスを与えている．
〔出力〕	コマンド実行後に戻されるレジスタ値などを示す．

(注1) 終了条件はＡＨに出力される
ＡＨ← 00H：正常
05H：不正呼び出し
06H：演算オーバーフロー
07H：作業領域不足で処理中断

(注2)〔 〕内は16色モード時

(1)初期化コマンド(♯INIT)

[機能]

グラフィックLIOの初期化を行う．グラフィックLIOの使用に際して，最初に当コマンドを必ず実行しておく．

①カラーパレットは，下記のように初期設定される．

パレット番号	カラーコード*	カラーコード**
0	0 (黒)	000 (黒)
1	1 (青)	00F (青)
2	2 (赤)	0F0 (赤)
3	3 (紫)	0FF (紫)
4	4 (緑)	F00 (緑)
5	5 (水色)	F0F (水色)
6	6 (黄色)	FF0 (黄色)
7	7 (白)	FFF (白)

パレット番号	カラーコード***
8	777 (灰色)
9	00A (暗い青)
10	0A0 (暗い赤)
11	0AA (暗い紫)
12	A00 (暗い緑)
13	A0A (暗い水色)
14	AA0 (暗い黄色)
15	AAA (暗い白)

＊8色/8色モード　＊＊8色/4096色モード　＊＊＊16色/4096色モード

③ 当コマンドにより，初期設定される内容を以下にまとめて示す.

フォアグランドカラー	パレット番号7
バックグランドカラー	パレット番号0
ボーダーカラー	カラーコード0(黒)
表示モード	カラー，640×200ドット
表示スイッチ	グラフィック表示有，普通描画
アクティブページ	0(ページ0のみ描画可)
ディスプレイページ	1(ページ0のみ表示)
アクティブページの描画領域	アクティブページ全体
パレットモード	0(8色／8色モード)

[コード]

INT　0A0H

[レジスタ]

DS← 60H：ワークエリアの設定（データセグメントのベースアドレス）

SS← 60H
SP← 200H
} スタックエリアの設定

リスト 4-12
サンプルプログラム

```
                     ; GLIO-SAMPLE (INITIALIZE)
                     ;       BY INT 0A0H
                     ;
                     CSEG
                     ORG 0H
                     ;
0000 B86000          MOV AX,60H
0003 8BD0            MOV DX,AX
0005 8ED0            MOV SS,AX
0007 B80002          MOV AX,200H
000A 8BE0            MOV SP,AX
000C CDA0            INT 0A0H
000E F4              HLT
                     ;
                     END
```

⑵画面モード設定コマンド(♯SCREEN)

［機能］

画面モード，画面スイッチ，アクティブ画面，ディスプレイ画面を設定する．BASICのSCREEN文と同じ機能です．

［コード］

INT　0A1H

［レジスタ］

ＤＳ←　60H：ワークエリアの設定（データセグメントのベースアドレス）

ＳＳ←　60H：スタックエリアの設定

ＳＰ←　200H ｝パラメータエリアの設定
ＢＸ←1500H

［パラメータ］

相対アドレス	パラメータ名
BX＋00H	画面モード
BX＋01H	画面スイッチ
BX＋02H	アクティブ画面
BX＋03H	ディスプレイ画面

①　画面モードのパラメータ値

パラメータ値	設定状態
00H	カラーグラフィックモード　(640×200)
01H	モノクログラフィックモード (640×200)
02H	高解像モノクロ　(640×400)
03H	高解像カラー　(640×400
FFH	現状の設定のまま

②　画面スイッチのパラメータ値

パラメータ値	設定状態
00H	グラフィック表示し，　高速書き込みせず
01H	グラフィック表示し，　高速書き込みする
02H	グラフィック表示せず，高速書き込みせず*
03H	グラフィック表示せず，高速書き込みする
FFH	現状の設定のまま

＊　E/F/Mでは高速書き込みする(03Hに同じ)

③アクティブ画面のパラメータ値（画面モードにより異なる）

画面コード	パラメータ値		G-VRAMの使用形態
	U	U以外	
カラー	0～1	0～3	2つに分割して使用
モノクロ	0～5(7)	0～11(15)	6(8)つに分割して使用
高解像カラー	0～2(3)	0～5(7)	3(4)つに分割して使用
高解像モノクロ	0	0～1	すべて使用

＊（　）の値は16色グラフィックモード時(16色グラフィックボードを実装する必要あり)
U以外で実装可能なのは，VF／VM．なお，UVは標準実装されている．

④ディスプレイ画面を指定するパラメータ値

(i)8色グラフィックモード時（拡張G-VRAMを使用しない）

ディスプレイ画面には，下記のように番号が割り当てられている．ただし，画面モードにより異なる．

画面番号	画面モード			
	カラー	モノクロ	高解像カラー	高解像モノクロ
1/7	$Pb1+Pr1+Pg1/\overline{Pb1}+\overline{Pr1}+\overline{Pg1}$	$PB1/\overline{PB1}$	$PB/\overline{PB}$	$Pb+Pr+Pg/\overline{Pb}+\overline{Pr}+\overline{Pg}$
2/8		$PR1/\overline{PR1}$	$PR/\overline{PR}$	
3/9		$PG1/\overline{PG1}$	$PG/\overline{PG}$	
4/10	$Pb2+Pr2+Pg2/\overline{Pb2}+\overline{Pr2}+\overline{Pg2}$	$PB2/\overline{PB2}$		
5/11		$PR2/\overline{PR2}$		
6/12		$PG2/\overline{PG2}$		

注）画面名称については，図4-2参照．
ディスプレイ画面の選択は，5ビットのパラメータで下記のように行う．

パラメータ値＼ビット	b_4	b_3	b_2	b_1	b_0
0 ～ 7	0	0	第3画面	第2画面	第1画面
8 ～ 15	0	1	第6画面	第5画面	第4画面
16 ～ 23	1	0	第9画面	第8画面	第7画面
24 ～ 31	1	1	第12画面	第11画面	第10画面

注）b_4 b_3はグループの選択スイッチ
注）b_2，b_1，b_0=1の画面が表示される．例えば，モノクロモード時，パラメータを00111（＝7）にすると，PB1，PR1，PG1が合成される．

(ii)16色グラフィックモード時（拡張G-VRAMを使用する）

ディスプレイ画面には，下記のように番号が割り当てられている。ただし，画面モードにより異なる．

画面番号	画面モード			
	カラー	モノクロ	高解像カラー	高解像モノクロ
1/9	$Pb1+Pr1+Pg1+Pi1/\overline{Pb1}+\overline{Pr1}+\overline{Pg1}+\overline{Pi1}$	$PB1/\overline{PB1}$	$PB/\overline{PB}$	$Pb+Pr+Pg+Pi/\overline{Pb}+\overline{Pr}+\overline{Pg}+\overline{Pi}$
2/10		$PR1/\overline{PR1}$	$PR/\overline{PR}$	
3/11		$PG1/\overline{PG1}$	$PG/\overline{PG}$	
4/12		$PI1/\overline{PI1}$	$PI/\overline{PI}$	
5/13	$Pb2+Pr2+Pg2+Pi2/\overline{Pb2}+\overline{Pr2}+\overline{Pg2}+\overline{Pi2}$	$PB2/\overline{PB2}$		
6/14		$PR2/\overline{PR2}$		
7/15		$PG2/\overline{PG2}$		
8/16		$PI2/\overline{PI2}$		

注）画面名称については，図4-2参照．

ディスプレイ画面の選択は，6ビットのパラメータで下記のように行う．

ビット / パラメータ値	b_5	b_4	b_3	b_2	b_1	b_0
0 ～ 7	0	0	第4画面	第3画面	第2画面	第1画面
8 ～ 15	0	1	第8画面	第7画面	第6画面	第5画面
16 ～ 23	1	0	第12画面	第11画面	第10画面	第9画面
24 ～ 31	1	1	第16画面	第15画面	第14画面	第13画面

注）$b_5 b_4$はグループの選択スイッチ

注）b_3, b_2, b_1, b_0=1の画面が表示される。

例えば，モノクロモード時，パラメータを111111（=63）にすると，PB2, PR2, PG2, PI2が合成表示される．

リスト 4-13

サンプルプログラム

```
                    ;  GLIO-SAMPLE (SCREEN)
                    ;        BY INT 0A1H
                    ;
                    CSEG
                    ORG 0H
                    ;
0000 B86000         MOV AX,60H
0003 8ED8           MOV DS,AX
0005 8ED0           MOV SS,AX
0007 B80002         MOV AX,200H
000A 8BE0           MOV SP,AX
000C BB0015         MOV BX,1500H
000F CDA1           INT 0A1H
0011 F4             HLT
                    ;
                    DSEG                        ; 60H
                    ORG 1500H
                    DATA:                       ; SCREEN 3,0,0,1
1500 03             MODE           DB 3
1501 00             SWITCH         DB 0
1502 00             ACTIVE         DB 0
1503 01             DISPLAY        DB 1
                    ;
                    END
```

⑶描画領域設定コマンド(#VIEW)

[機能]

アクティブ画面における描画領域（ビューポート）を設定し，ビューポート内を塗りつぶして，外枠を描く．

[コード]

INT　0A2H

[レジスタ]

DS←60H　:ワークエリアの設定（データセグメントのベースアドレス）

SS←60H
SP←200H　} スタックエリアの設定

BX←1500H:パラメータエリアの設定

[パラメータ]

相対アドレス	パラメータ名
BX＋00H，＋01H	X1，ビューポート左上のX座標
BX＋02H，＋03H	Y1，ビューポート左上のY座標
BX＋04H，＋05H	X2，ビューポート右下のX座標
BX＋06H，＋07H	Y2，ビューポート右下のY座標
BX＋08H	領域色 {00H～07H(パレット番号）または FFH（塗りつぶさない）
BX＋09H	境界色 {00H～07H(パレット番号）または FFH（外枠を描かない）

(4)背景色設定コマンド(#COLOR1)

[機能]

バックグラウンドカラー，ボーダーカラー，フォアグラウンドカラーを設定.

[コード]

INT　0A3H

[レジスタ]

DS←60H　：ワークエリアの設定（データセグメントのベースアドレス）

SS←60H　：スタックエリアの設定

SP←200H ：

BX←1500H：パラメータエリアの設定

[パラメータ]

相対アドレス	パラメータ名	
BX+00H	未使用	
BX+01H	バックグラウンドカラー	00H~07H〔0FH〕(パレット番号) またはFFH（現状の設定のまま）
BX+02H	ボーダーカラー	00H~07H(カラーコード)または FFH（現状の設定のまま）
BX+03H	フォアグラウンドカラー	00H~07H〔0FH〕(パレット番号) またはFFH

リスト 4-14
サンプルプログラム

```
                        ;  GLIO-SAMPLE  (COLOR1)
                        ;       BY INT 0A3H
                        ;
                        CSEG
                        ORG 0H
                        ;
0000 B86000             MOV AX,60H
0003 8ED8               MOV DS,AX
0005 8ED0               MOV SS,AX
0007 B80002             MOV AX,200H
000A 8BE0               MOV SP,AX
000C BB0015             MOV BX,1500H
000F CDA3               INT 0A3H
0011 F4                 HLT
                        ;
                        DSEG                        ; 60H
                        ORG 1500H
                        DATA:                       ; COLOR ,2,0,7
1500 00                 UN_USE          DB 0
1501 02                 BACK_C          DB 2
1502 00                 BORDER_C        DB 0
1503 07                 FOR_C           DB 7
1504 00                 MODE            DB 0
                        ;
                        END
```

⑸パレットレジスタ設定コマンド(#COLOR2)

[機能]

パレット番号とカラーコードの対応関係を設定する.

[コード]

INT 0A4H

[レジスタ]

DS←60H :ワークエリアの設定（データセグメントのベースアドレス)

SS←60H
SP←200H } スタックエリアの設定

BX←1500H:パラメータエリアの設定

[パラメータ]

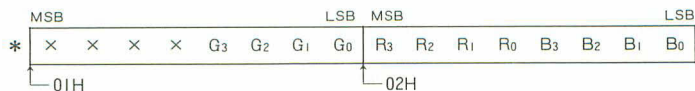
パレットモードにより異なる.

(i) 8色／8色モード時

相対アドレス	パラメータ名
BX + 00H	パレット番号（00H～07H)
BX + 01H	カラーコード（00H～07H)

(ii) 8色／4096色モード，16色／4096色モード時

相対アドレス	パラメータ名
BX +00H	パレット番号 00H～07H：8色／4096色モード 00H～0FH：16色／4096色モード
BX +01H, +02H	カラーコード*

*	MSB							LSB	MSB							LSB
	×	×	×	×	G_3	G_2	G_1	G_0	R_3	R_2	R_1	R_0	B_3	B_2	B_1	B_0
	01H								02H							

リスト 4-15
サンプルプログラム

```
                                        ;  GLIO-SAMPLE (COLOR2)
                                        ;        BY INT 0A4H
                                        ;
                                        CSEG
                                        ORG 0H
                                        ;
0000 B86000                             MOV AX,60H
0003 8ED8                               MOV DS,AX
0005 8ED0                               MOV SS,AX
0007 B80002                             MOV AX,200H
000A 8BE0                               MOV SP,AX
000C BB0015                             MOV BX,1500H
000F B90800                             MOV CX,8
                                        LOOP1:
0012 53                                 PUSH BX
0013 51                                 PUSH CX
0014 CDA4                               INT 0A4H ―――――― #COLOR2
0016 59                                 POP CX
0017 5B                                 POP BX
0018 43                                 INC BX
0019 43                                 INC BX
001A E2F6            0012               LOOP LOOP1
001C F4                                 HLT
                                        ;
                                        DSEG                        ; 60H
                                        ORG 1500H
                                        DATA:                       ; COLOR=(I,I)_I=0-7
1500 00                                 PALETTE         DB 0
1501 00                                 C_CODE          DB 0
1502 01                                 PALETTE1        DB 1
1503 01                                 C_CODE1         DB 1
1504 02                                 PALETTE2        DB 2
1505 02                                 C_CODE2         DB 2
1506 03                                 PALETTE3        DB 3
1507 03                                 C_CODE3         DB 3
1508 04                                 PALETTE4        DB 4
1509 04                                 C_CODE4         DB 4
150A 05                                 PALETTE5        DB 5
150B 05                                 C_CODE5         DB 5
150C 06                                 PALETTE6        DB 6
150D 06                                 C_CODE6         DB 6
150E 07                                 PALETTE7        DB 7
150F 07                                 C_CODE7         DB 7
                                        ;
                                        END
```

⑹クリアコマンド(#CLS)

[機能]

アクティブ画面における描画領域をバックグラウンドカラーで塗りつぶす.

[コード]

INT　0A5H

[レジスタ]

ＤＳ←60H　：ワークエリアの設定（データセグメントのベースアドレス）

ＳＳ←60H
ＳＰ←200H　｝スタックエリアの設定

⑺プロットコマンド(#PSET)

[機能]

アクティブ画面における指定座標に指定色の点を描く.

[コード]

INT　0A6H

[レジスタ]

ＤＳ←60H　：ワークエリアの設定

ＳＳ←60H
ＳＰ←200H　｝スタックエリアの設定

ＢＸ←1500H：パラメータエリアの設定

AH：動作モードの指定

[パラメータ]

相対アドレス	パラメータ名
BX＋00H, ＋01H	X座標
BX＋02H, ＋03H	Y座標
BX＋04H	パレット番号 ｛00H～07H〔0FH〕(パレット番号）または FFH（現状の設定のまま）

AH	パレット番号FFH指定のとき
01H	フォアグラウンドカラーのパレット番号を使う
02H	バックグラウンドカラーのパレット番号を使う

(8)ラインコマンド(#LINE)

[機能]

指定した2点間を結ぶ直線，またはこの直線を対角線とする矩形を描く．

[コード]

INT　0A7H

[レジスタ]

ＤＳ←60H　：ワークエリアの設定

ＳＳ←60H
ＳＰ←200H　｝スタックエリアの設定

ＢＸ←1500H：パラメータエリアの設定

[パラメータ]

相対アドレス	パタメータ名
BX＋00H，＋01H BX＋02H，＋03H BX＋04H，＋05H BX＋06H，＋07H	X1：始点のＸ座標 Y1：始点のＹ座標 X2：終点のＸ座標 Y2：終点のＹ座標
BX＋08H	線指定色 ｛00H～07H〔0FH〕(パレット番号）または FFH（フォアグラウンドカラー）
BX＋09H	描画コード ｛00H：直線 01H：矩形 02H：矩形を塗りつぶす
BX＋0AH	ラインスタイル，矩形塗りつぶし色，タイルパターンの指定スイッチ ｛00H：何も指定しない 01H：ラインスタイル，塗りつぶし色の指定あり 02H：タイルパターンの指定あり
BX＋0BH	塗りつぶし色：00H～07H〔0FH〕(パレット番号） または，ラインスタイルL（ラインスタイル下位8ビットのパターン） （注）｛描画コードが00H又は01Hの時はラインスタイルL 描画コードが02Hの時は塗りつぶし色
BX＋0CH BX＋0DH BX＋0EH，＋0FH BX＋10H，＋11H	ラインスタイルH（ラインスタイル上位８ビットのパターン） タイルパターン長：00H～07H（描画コード02Hの時のみ有効） タイルパターン格納域先頭アドレス（オフセットアドレス） タイルパターン格納域先頭アドレス（セグメントアドレス）

リスト 4-16
サンプルプログラム

```
                              ;  GLIO-SAMPLE (LINE)
                              ;        BY INT 0A7H
                              ;
                              CSEG
                              ORG 0H
                              ;
 0000 B86000                  MOV AX,60H
 0003 8ED8                    MOV DS,AX
 0005 8ED0                    MOV SS,AX
 0007 B80002                  MOV AX,200H
 000A 8BE0                    MOV SP,AX
 000C BB0015                  MOV BX,1500H
 000F CDA7                    INT 0A7H ──────────── #LINE
 0011 F4                      HLT
                              ;
                              DSEG                       ; 60H
                              ORG 1500H
                              DATA :                     ;LINE(100,50)-(400,200),
2,BF
 1500 6400                    X1            DW 100
 1502 3200                    Y1            DW 50
 1504 9001                    X2            DW 400
 1506 C800                    Y2            DW 200
 1508 05                      COLOR_P1      DB 5
 1509 02                      CODE          DB 2         ; --+
 150A 01                      SWITCH        DB 1         ; --!
 150B 02                      COLOR_P2      DB 2         ;<--*
 150C 00                      LINE_ST_H     DB 0
 150D 00                      TILE_LNG      DB 0
 150E 0000                    TILE_PAT_OFF  DW 0
 1510 0000                    TILE_PAT_SEG  DW 0
                              ;
                              END
```

(9)サークルコマンド(#CIRCLE)

[機能]

中心座標，X方向半径，Y方向半径を指定し，円または楕円を描く．あるいは，開始点，終了点を指定し，円弧または扇形を描く．

[コード]

INT　0A8H

[レジスタ]

DS←60H　：ワークエリアの設定

SS←60H
SP←200H　} スタックエリアの設定

BX←1500H：パラメータエリアの設定

[パラメータ]

相対アドレス	パラメータ名
BX＋00H，＋01H	中心のX座標
BX＋02H，＋03H	中心のY座標
BX＋04H，＋05H	X方向半径
BX＋06H，＋07H	Y方向半径
BX＋08H	描画色指定 {00H～07H〔0FH〕(パレット番号)または FFH (フォアグラウンドカラー)
BX＋09H	フラグ　(注)
BX＋0AH，＋0BH	開始点のX座標
BX＋0CH，＋0DH	開始点のY座標
BX＋0EH，＋0FH	終了点のX座標
BX＋10H，＋11H	終了点のY座標
BX＋12H	タイルパターン指定時には，タイルパターンデータ長：00～FFH タイルパターン未指定時には，塗りつぶし色指定 {00H～07H〔0FH〕(パレット番号)または FFH (描画色と同じ)
BX＋13H，＋14H	タイルパターン格納域先頭アドレス(オフセットアドレス)
BX＋15H，＋16H	タイルパターン格納域先頭アドレス(セグメントアドレス)

(注)

bit	内容
bit 0	開始点指示の有無(0：なし，1：あり)
1	開始線分指示の有無(0：なし，1：あり)
2	終了点指示の有無(0：なし，1：あり)
3	終了線分指示の有無(0：なし，1：あり)
4	開始点，終了点一致時の処理指定(0：全楕円描画，1：1点のみ)
5	ぬりつぶしの指示(0：なし，1：ぬりつぶし)
6	タイルパターン指示の有無(0：なし，1：あり)

リスト 4-17
サンプルプログラム

```
                              ;  GLIO-SAMPLE (CIRCLE)
                              ;       BY INT 0A8H
                              ;
                              CSEG
                              ORG 0H
                              ;
0000 B86000                   MOV AX,60H
0003 8ED8                     MOV DS,AX
0005 8ED0                     MOV SS,AX
0007 B80002                   MOV AX,200H
000A 8BE0                     MOV SP,AX
000C BB0015                   MOV BX,1500H
000F CDA8                     INT 0A8H ------------  #CIRCLE
0011 F4                       HLT
                              ;
                              DSEG                          ; 60H
                              ORG 1500H
                              DATA:
1500 C800                     CIRX              DW 200     ;CHUSIN
1502 C800                     CIRY              DW 200
1504 9600                     RX                DW 150     ;HANKEI
1506 6400                     RY                DW 100
1508 04                       COLOR_P1          DB 4
1509 2F                       FLAG              DB 2FH
150A C800                     SX                DW 200     ;KAISI-TEN
150C 6400                     SY                DW 100
150E 5E01                     EX                DW 350     ;OWARI-TEN
1510 C800                     EY                DW 200
1512 03                       COLOR_P2          DB 3
1513 0000                     TILE_PAT_OFF      DW 0
1515 0000                     TILE_PAT_SEG      DW 0
                              ;
                              END
```

⑽ペイントコマンド（#PAINT1）

［機能］

指定した点と指定境界色で決定される領域を指定色で塗りつぶす．

［コード］

INT　0A9H

［レジスタ］

ＤＳ←60H　：ワークエリアの設定

ＳＳ←60H
ＳＰ←200H　｝スタックエリアの設定

ＢＸ←1500H：パラメータエリアの設定

［パラメータ］

相対アドレス	パラメータ名
BX＋00H，＋01H	塗りつぶし開始点のＸ座標
BX＋02H，＋03H	塗りつぶし開始点のＹ座標
BX＋04H	領域色指定 {00H～07H〔0FH〕(パレット番号)または FFH（フォアグラウンドカラー）
BX＋05H	境界色指定 {00H～07H〔0FH〕（パレット番号)または FFH（領域色と同じに設定）
BX＋06H，＋07H	作業域の最終アドレス（オフセットアドレス）　（注）
BX＋08H，＋09H	作業域の先頭アドレス（オフセットアドレス）

（注）16バイト以上の作業域が必要（この領域をユーザは使用してはいけない．ＤＳ内に存在）

(11)タイルパターンペイントコマンド(#PAINT2)

[機能]

指定した点と指定境界色で決定される領域を指定のタイルパターンで塗りつぶす.

[コード]

INT　0AAH

[レジスタ]

DS←60H　：ワークエリアの設定

SS←60H
SP←200H　｝スタックエリアの設定

BX←1500H：パラメータエリアの設定

[パラメータ]

相対アドレス	パラメータ名
BX+00H, +01H	塗りつぶし開始点のX座標
BX+02H, +03H	塗りつぶし開始点のY座標
BX+04H	未使用
BX+05H	タイルパターン長
BX+06H, +07H	タイルパターン格納域の先頭アドレス（オフセットアドレス）
BX+08H, +09H	タイルパターン格納域の先頭アドレス（セグメントアドレス）
BX+0AH	境界色指定 ｛00H～07H〔0FH〕（パレット番号）または FFH（領域色と同じに設定）
BX+10H, +11H	作業域の最終アドレス（オフセットアドレス）　（注）
BX+12H, +13H	作業域の先頭アドレス（オフセットアドレス）

（注）16バイト以上の作業域を確保しておく必要がある.

注）タイルパターンは，タイリングに用いられる基本タイルの模様と大きさを決定する文字列である．タイルの大きさは，横方向は8ドット分と決められているが，縦方向の長さはタイルパターン長で指定することができる.

縦方向がnドットのタイルを指定するためには，モノクロモードでn文字，カラーモードで3×n文字(16色／4096色モードの場合は4×n文字)の長さを必要とする．タイルパターンの指定の方法はカラーモードとモノクロモードとで異なり，モノクロモードでは，1バイトを横8ビットの線に対応させて指定し，カラーモードでもモノクロモードと同じように，模様はタイルパターンに対応するビットパターンによって決定されるが，モノクロモードと異なり，3バイト（16色／4096色モードのばあいは4バイト）で横8ドットが構成される．タイルパターン中の文字はドットにどのパレット番号を対応させるかを決定する．タイルパターンの長さに余りがあった場合には無視されるが，3文字(あるいは4文字)に満たない場合にはエラーとなる.

リスト 4-18
サンプルプログラム

```
                          ;  GLIO-SAMPLE (TILE-PAINT)
                          ;       BY INT 0AAH
                          ;
                          CSEG
                          ORG 0H
                          ;
0000 B86000               MOV AX,60H
0003 8ED8                 MOV DS,AX
0005 8ED0                 MOV SS,AX
0007 A30815               MOV TILE_PAT_SEG,AX
000A B80002               MOV AX,200H
000D 8BE0                 MOV SP,AX
000F B81415               MOV AX,OFFSET TILE_DATA
0012 A30615               MOV TILE_PAT_OFF,AX
0015 8D064215             LEA AX,EN_WORK
0019 A31015               MOV WORK_EN_OFF,AX
001C 8D061A15             LEA AX,ST_WORK
0020 A31215               MOV WORK_ST_OFF,AX
0023 BB0015               MOV BX,1500H
0026 CDAA                 INT 0AAH ──────────── #PAINT
0028 F4                   HLT
                          ;
                          DSEG                        ; 60H
                          ORG 1500H
                          DATA:                       ; PAINT(100,100),TILE$,7
1500 6400                 X               DW 100
1502 6400                 Y               DW 100
1504 00                   UN_USE1         DB 0
1505 06                   TILE_LNG        DB 6
1506 0000                 TILE_PAT_OFF    DW 0
1508 0000                 TILE_PAT_SEG    DW 0
150A 07                   COLOR_P         DB 7
150B 0000000000           UN_USE2         DB 0,0,0,0,0
1510 0000                 WORK_EN_OFF     DW 0
1512 0000                 WORK_ST_OFF     DW 0
                          ;
                          TILE_DATA:────────────タイルパターンデータ
1514 00AAFF0055FF         DA              DB 00H,0AAH,0FFH,00H,55H,0FFH
                          ;
151A                      ST_WORK         RW 20
1542                      EN_WORK         RW 1
                          ;
                          END
```

⑿描画情報検出コマンド(#GET)

[機能]

指定領域の描画情報を指定した格納域へ格納する.

BASICののGET@(X1, Y1) - (X2, Y2), <配列名>と同じ機能です.

[コード]

INT　0ABH

[レジスタ]

DS←60H　：ワークエリアの設定

SS←60H
SP←200H　} スタックエリアの設定

BX←1500H：パラメータエリアの設定

[パラメータ]

相対アドレス	パラメータ名
BX+00H, +01H	指定領域左上点のX座標：X1
BX+02H, +03H	指定領域左上点のY座標：Y1
BX+04H, +05H	指定領域右下点のX座標：X2
BX+06H, +07H	指定領域右下点のY座標：Y2
BX+08H, +09H	格納域先頭アドレス（オフセットアドレス）
BX+0AH, +0BH	格納域先頭アドレス（セグメントアドレス）
BX+0CH, +0DH	格納域の長さ（単位：バイト）　（注）

（注）格納域の長さ

≧((X2−X1+8)¥8)・(Y2−Y1+1)・A+4

ただしA＝{ 1：白黒モード
3：8色カラーモード
4：16色カラーモード

リスト 4-19
サンプルプログラム

```
                          ;   GLIO-SAMPLE(GGET)
                          ;       BY INT 0AAH
                          ;
                          CSEG
                          ORG 0H
                          ;
0000 B86000               MOV AX,60H
0003 8ED8                 MOV DS,AX
0005 8ED0                 MOV SS,AX
0007 A30A15               MOV WORK_SEG,AX
000A B80002               MOV AX,200H
000D 8BE0                 MOV SP,AX
000F 80060E15             LEA AX,WORK_ST
0013 A30815               MOV WORK_OFF,AX
0016 BB0015               MOV BX,1500H
0019 CDAB                 INT 0ABH ─────────── #GET
001B F4                   HLT
                          ;
                          DSEG                    ; 60H
                          ORG 1500H
                          DATA:
1500 6400                 X1            DW 100
1502 3200                 Y1            DW 50
1504 7800                 X2            DB 120
1506 4600                 Y2            DW 70
1508 0000                 WORK_OFF      DW 0
150A 0000                 WORK_SEG      DB 0
150C C800                 WORK_LNG      DW 200
                          ;
150E                      WORK_ST       RW 100
15D6                      WORK_EN       RW 1
                          ;
                          END
```

⒀描画情報読み取りコマンド（#PUT1）

［機能］

指定格納域の画像情報に基づいて，指定領域を描画する．

BASICのPUT＠（X1，Y1），<配列名>と同じ機能です．

［コード］

INT　0ACH

［レジスタ］

DS←60H　：ワークエリアの設定

SS←60H
SP←200H　｝スタックエリアの設定

BX←1500H：パラメータエリアの設定

［パラメータ］

相対アドレス	パラメータ名
BX＋00H，＋01H	描画領域左上点のX座標
BX＋02H，＋03H	描画領域左上点のY座標
BX＋04H，＋05H	格納域先頭アドレス（オフセットアドレス）
BX＋06H，＋07H	格納域先頭アドレス（セグメントアドレス）
BX＋08H，＋09H	格納域の長さ（単位：バイト）
BX＋0AH	描画モード指定　（注1）
BX＋0BH	カラースイッチ　（注2）
BX＋0CH	フォアグラウンドカラー：00H～07H〔0FH〕（パレット番号）
BX＋0DH	バックグラウンドカラー：00H～07H〔0FH〕（パレット番号）

（注1）

描画モード	描画規則
00H	PSET
01H	PRESET
02H	OR
03H	AND
04H	XOR

（注2）

カラースイッチ	説明
00H	フォアグラウンドカラー，バックグラウンドカラーを指定せず現在の描画モードを維持
01H	フォアグラウンドカラー，バックグラウンドカラーを指定するモノクロモードで描画

⒁日本字描画コマンド(#PUT2)

[機能]

指定の日本語(JISコードで指定)を指定領域上に描画する.

[コード]

INT　0ADH

[レジスタ]

DS←60H　：ワークエリアの設定

SS←60H
SP←200H　｝スタックエリアの設定

BX←1500H：パラメータエリアの設定

[パラメータ]

相対アドレス	パラメータ名
BX+00H, +01H	描画領域左上点のX座標
BX+02H, +03H	描画領域左上点のY座標
BX+04H, +05H	日本語JISコード
BX+06H	描画モード　｝⒀項参照
BX+07H	カラースイッチ　｝⒀項参照
BX+08H	フォアグラウンドカラー：00H～07H(パレット番号)
BX+09H	バックグラウンドカラー：00H～07H(パレット番号)

リスト 4-20
サンプルプログラム

```
                        ;  GLIO-SAMPLE (KNJ-PUT)
                        ;        BY INT 0ADH
                        ;
                        CSEG
                        ORG 0H
                        ;
0000 B86000             MOV AX,60H
0003 8ED8               MOV DS,AX
0005 8ED0               MOV SS,AX
0007 B80002             MOV AX,200H
000A 8BE0               MOV SP,AX
000C BB0015             MOV BX,1500H
000F CDAD               INT 0ADH
0011 F4                 HLT
                        ;
                        DSEG                        ; 60H
                        ORG 1500H
                        DATA:
1500 6400               X                 DW 100
1502 6400               Y                 DW 100
1504 3333               KNJ_CODE          DW 3333H
1506 00                 MODE              DB 0
1507 00                 SWITCH            DB 0
1508 00                 FOR_C             DB 0
1509 00                 BACK_C            DB 0
                        ;
                        END
```

⒂描画画面スクロールコマンド(#ROLL)

[機能]

アクティブ画面全体の描画情報を指定ドット数分だけ上下左右へ移動する．

[コード]

INT　0AEH

[レジスタ]

ＤＳ←60H　：ワークエリアの設定

ＳＳ←60H
ＳＰ←200H　} スタックエリアの設定

ＢＸ←1500H：パラメータエリアの設定

[パラメータ]

相対アドレス	パラメータ名
BX＋00H, ＋01H	上下方向移動量（－399～399：ドット）
BX＋02H, ＋03H	左右方向移動量（－639～639：ドット）(注)
BX＋04	クリアフラグ {00H：移動の残領域をパレット0で塗る {01H：移動の残領域をバックグラウンドカラーで塗る

(注) 左右方向へ実際に移動するドット数は，その絶対値以下で最も近い８の倍数分である。

リスト4-21
サンプルプログラム

```
                        ;     GLIO-SAMPLE (ROLL)
                        ;        BY INT 0AEH
                        ;
                        CSEG
                        ORG 0H
                        ;
0000 B86000             MOV AX,60H
0003 8ED8               MOV DS,AX
0005 8ED0               MOV SS,AX
0007 B80002             MOV AX,200H
000A 8BE0               MOV SP,AX
000C BB0015             MOV BX,1500H
000F CDAE               INT 0AEH
0011 F4                 HLT
                        ;
                        DSEG      ;60H
                        ORG 1500H
                        DATA:
1500 6400               UP_DOWN          DW 100
1502 6400               LEFT_RIGHT       DW 100
1504 01                 CLEAR_FLAG       DB 1
                        ;
                        END
```

⒃ドットの色情報検出コマンド(#POINT)

[機能]

指定座標のドットの色情報（パレット番号）を検出する．

[コード]

INT　0AFH

[レジスタ]

ＤＳ←60H　:ワークエリアの設定（データセグメントのベースアドレス）

ＳＳ←60H
ＳＰ←200H
}スタックエリアの設定

ＢＸ←1500H:パラメータエリアの設定

[パラメータ]

相対アドレス	パラメータ名
BX＋00H，＋01H	指定ドットのＸ座標
BX＋02H，＋03H	指定ドットのＹ座標

[出力]

AL←パレット番号

ALの値	
FFH	指定座標がアクティブ画面のビューポート以外
00H～07H	画面モードがカラー：パレット番号を示す
00Hか01H	画面モードがモノクロ：00H(黒）／01H（白）

(17)領域内ドット色情報検出コマンド(#COPY)

[機能]

現在表示されている画面上の指定領域におけるドット状態に関する情報を指定格納域へ格納する．

[コード]

INT　0CEH

[レジスタ]

ＤＳ←60H　：ワークエリアの設定（データセグメントのベースアドレス）

ＳＳ←60H
ＳＰ←200H　｝スタックエリアの設定

ＡＸ←指定領域左上点のx座標［X］

ＢＸ←指定領域左上点のy座標［Y］

ＣＬ←指定領域x方向長さ［Xd］（単位：ドット）

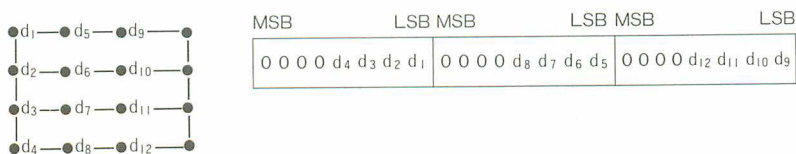
ＣＨ←指定領域y方向長さ［Yd］（単位：ドット）(02H, 04H, 08H, 82H, 84H)

ＤＩ←格納域の先頭アドレス（オフセットアドレス）

ＥＳ←格納域の先頭アドレス（セグメントアドレス）

（注）X，Xdは８の倍数

（注）Yd＝04Hの場合，各ドットdiの情報は次のように格納される．

MSB　　LSB	MSB　　LSB	MSB　　LSB
0 0 0 0 d_4 d_3 d_2 d_1	0 0 0 0 d_8 d_7 d_6 d_5	0 0 0 0 d_{12} d_{11} d_{10} d_9

ただし

カラーモード時 ｛ドットのパレット番号が0　→ｄｉ＝0
　　　　　　　　ドットのパレット番号が0以外→ｄｉ＝1

モノクロモード時 ｛ドットが黒　→ｄｉ＝0
　　　　　　　　　ドットが白　→ｄｉ＝1

8 グラフィックチャージャー

グラフィックチャージャ(GRCG)は，CPUがG-VRAMにデータを書き込んだり，あるいはG-VRAMからデータを読み出すときに，間に入ってデータを加工（論理演算）するハードウェアです．描画データに対して論理演算するための基本データは，1バイト長のタイル・レジスタにあらかじめ設定しておきます．タイルレジスタは，各プレーン（B,R,G,Iプレーン）に独立に1個ずつ設けられています．

(CRCGを持つ機種はUV／VM／VF／VX／UX．Uはオプション)

8.1 グラフィックチャージャの I/O制御命令

グラフィックチャージャは，3つの動作モード（TDWモード，TCRモード，RMWモード）を持っています．動作モードの選択やタイル・レジスタの設定を行うためのI／O制御命令を**表4-15**に示します．

表4-15 グラフィックチャージャのI／O制御命令

I／O制御命令	I／Oポートアドレス	I／O	制御データ								機能説明
			b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0	
ライト モードレジスタ	7CH	OUT	CG	RMW	0	0	PI	PG	PR	PB	動作モードの設定および有効プレーン(B,R,G,I)の指定
ライト タイルレジスタ	7EH	OUT	← タイルパターンデータ →								タイルレジスタB,R,G,Iの値を設定する

(注)

ビット名	ビット値＝1	ビット値＝0
CG	GRCGを有効にする (CPUがG-VRAMにアクセスした時，GRCGが動作する)	GRCRを無効にする
RMW	RMWモード(G-VRAMライト時) (注) G-VRAMリードは無視する	TDWモード(G-VRAMライト時) TCRモード(G-VRAMリード時)
PB,PR PG, PI	B,R,G,Iプレーンを無効にする	B,R,G,Iプレーンを有効にする

(注) タイルレジスタの指定について
ライトモードレジスタ命令の実行により，タイルレジスタBが選択され，以後，ライトタイルレジスタ命令を実行する毎に，タイルレジスタR，G，Iの順に切りかわる．

≡8.2≡ グラフィックチャージャの動作モード

グラフィックチャージャ（GRCG）の動作モード，TDW，TCR，RMWモードについて解説します．

(1)TDWモード

CRCGがTDWモードに設定されていると，CPUがG-VRAMにデータを書き込むときCPUのデータは無視され，代わりにCRCGのタイルレジスタにセットされているデータが書き込まれます．このモードを使うと，CPUが１度書き込むだけで最大４プレーンに同時書き込みできます．

なお，CPUでワードアクセスする場合には，１バイトのタイルレジスタの値が１ワードに拡張されて使用されます．

(2)TCRモード

CRCGがTCRモードに設定されていると，GRCGはCPUがG-VRAMから読み出すデータを加工します．GRCGは，有効なプレーン毎にプレーンの値とタイルレジスタとをXORし，さらにNOTします．最後に，全ての有効なプレーンの演算結果のANDをとった値をCPUに返します．つまりあるビットについて，タイルレジスタの値と，有効な全プレーンの値がすべて一致したときに，CPUの読み出すデータの該当ビットが１になるのです．

(3)RMWモード

GRCGがRMWモードに設定されていると，CPUがG-VRAMにデータを書き込むときに，以下に示す規則にしたがって論理演算を施したデータを書き込みます．CPUが書き込もうとするデータの"1"のビットの部分は，タイルレジスタの該当ビットの値で置き換えられます．CPUが書き込もうとするデータの"0"のビットの部分は，変化しません．

```
1 1 1 1 0 0 0 0 ←CPUが書き込もうとするデータ
0 1 0 1 0 1 0 1 ←G-VRAMの旧データ
0 0 1 1 0 0 1 1 ←タイルレジスタ
   ↓       ↓
0 0 1 1 0 1 0 1 ←G-VRAMの新データ
```

第5章

フロッピーディスク

CONTENT

1 概要

フロッピーディスク装置は，PC-98の外部記憶装置として，とても重要な構成要素です．それだけに，記録密度，処理速度などの性能は，著しい向上の一途をたどっています．PC-98には，多種のバージョンがありますが，各バージョンの差異を特長づけているのが，標準実装されているフロッピーディスク装置の差異であるとさえ言えるほどです．

本章では，バージョン毎に標準実装されているフロッピーディスク装置がどのように異なっているかを簡単に説明するとともに，フロッピーディスク装置でデータ管理する場合の重要な概念であるところのファイル管理についても解説します．

さらに，フロッピーディスク装置を効率よく，しかも，より簡単な手続きで制御するために用意されている基本ソフトウェア（DISK BIOS，DISK LIO）についても解説しています．

2 フロッピーディスク

2.1 フロッピーディスクの物理構造

(1)フロッピーディスク装置の種類

PC-98で標準実装されているフロッピーディスク装置の種類を**表5-1**にまとめて示します．

フロッピーディスクの形状から分類すると，3.5インチ，5インチ，8インチの3種類があります．記録密度から分類すると，倍密度，高密度の2種類があります．この場合の記録密度というのは，円周方向に関してのものです．これに対して，直径方向の記録密度を向上させた倍トラックタイプのものも出現しました．高密度化の実現により，5インチのディスクが従来の8インチディスクと同等の容量にまで性能が向上しています．

表5-1　フロッピーディスク装置の種類

装置名称	略称*1	容量	備　考
3.5インチ両面倍密度倍トラック マイクロフロッピーディスク装置	3.5"2DD FDD	640KB	PC-9801U2に標準実装
5インチ両面倍密度倍トラック ミニフロッピーディスク装置	5"2DD FDD	640KB	PC-9801F/VFに標準実装
3.5インチ高密度 マイクロフロッピーディスク装置	3.5"2HD FDD	1MB	PC-9801UVに標準実装 *2
5インチ両面倍密度 ミニフロッピーディスク装置	5"2D FDD	320KB	PC-98には標準実装されていない
5インチ高密度 ミニフロッピーディスク装置	5"2HD FDD	1MB	PC-9801M/VM/VXに標準実装*2
8インチ倍密度 フロッピーディスク装置	8"2D FDD	1MB	PC-98には標準実装されていない.

＊1　2DD＝両面倍密度倍トラック/2HD＝両面高密度倍トラック/2D＝両面倍密度
＊2　PC-9801VM, UVには，2DD/2HDのいずれにも切り替え可能なFDDが標準実装されている．

(2)フロッピーディスクの物理アドレス

ここでは，フロッピーディスクの物理アドレスについて説明します．

フロッピーディスクの記録面は，基本記憶容量を単位として規則的に分割されていて，個々の領域には物理アドレスが割り当てられています．

物理アドレスは，サーフェス番号（ヘッド番号），トラック番号（シリンダ番号），セクタ番号（レコード番号）の3階層で構成されています．サーフェス番号は，ディスクの面を指定します．

次に，ディスク面を同心円状の帯状領域に分割して考えたとき，個々の帯状領域をトラックと呼びます．外側のトラックから順番にトラック番号が割り当てられています．さらに，ディスク面を扇形領域に分割してできる個々の小さな領域をセクタと呼びます．ディスクへのRead/Writeは一般にセクタを基本単位として行われます．物理アドレスの範囲は，ディスク装置の種類によって異なります．それをまとめて**表5-2**に示します．

図5-1
5"2DD, 3.5"2DD
フロッピーディスクの
物理アドレス

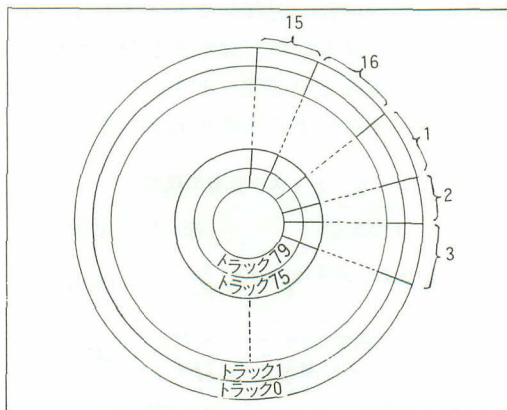



表5-2　物理アドレス（N88BASICのとき）

FD装置名	物理アドレス			セクタ当り容量	記録容量
	サーフェス番号	トラック番号	セクタ番号		
3.5"2DD	0〜1	0〜79	1〜16	256B	655.36KB
5"2DD	0〜1	0〜79	1〜16	256B	655.36KB
3.5"／5"2HD	0〜1	0〜76	1〜26	256B*	1.025MB
8"2D	0〜1	0〜76	1〜26	256B*	1.025MB

*サーフェス0，トラック0の部分だけは128B/セクタ

また，5インチ2DD，3.5インチ2DDフロッピーディスクの場合を例に取って，その物理アドレスとディスク面上の実際の位置との対応関係を**図5-1**に示します．なお，3.5インチ2HD，5インチ2HD，8インチ2Dフロッピーディスクについても，ほぼ**図5-1**と同様です．

2.2 フロッピーディスクのファイル管理(N88-DISK BASIC)

N88-DISK BASICシステムでは，フロッピーディスクのデータはファイルという概念に基づいて入出力管理されています．ファイルとは，データの集合に名称（ファイル名）を付与したものと考えることができます．以下では，N88-DISK BASICにおけるファイル管理方法について説明します．

(1)システムディスク

N88-DISK BASICのシステムディスクの使用状況を**表5-3**に示します．システム制御に関係した領域はIPL*，DISK CODE**，ディレクトリ，FAT，ディスクIDであり，これ以外がユーザ領域となります．IPL，DISK CODEは，ファイル管理とは直接関係ないので，ここでは説明を省略します．ディレクトリ，FAT，ディスクIDについては以降で説明します．

表5-3(1)　N88-DISKのシステムディスクの使用状況(3.5"2DD，5"2DD)

サーフェス番号	トラック番号	セクタ番号	内容
0	0	1～2	IPL
	0～8		DISK CODE
	9～39		ユーザ領域
	40	1～12	ディレクトリ
		13	ディスクID
		14～16	FAT
	41～79		ユーザ領域
1	0～8		DISK CODE
	9～79		ユーザ領域

*　IPL＝Initial Program Loader
DISK用OS(例えばDISK BASIC，CP/M，MS-DOS etc.)等をディスクからメモリへロードするための機械語プログラム

**　N88-DISK BASICが格納されている領域

表5-3(2)　N_{88}-DISK BASICのシステムディスクの使用状況（3.5"2HD, 5"2HD, 8"2D）

サーフェス番号	トラック番号	セクタ番号	内容
0	0	1～4	IPL
		5～26	システム予約
	1～9		DISK CODE
	10～34		ユーザ領域
	35	1～22	ディレクトリ
		23	ディスクID
		24～26	FAT
	36～76		ユーザ領域
1	0		システム予約
	1～9		DISK CODE
	10～76		ユーザ領域

(2)クラスタ

N_{88}-DISK BASICでは，ディスクデータのファイル管理を行ううえで，都合のよい記憶単位としてクラスタという概念を用いています．物理アドレスとクラスタ番号の対応関係を表5-4に示します．クラスタ番号に対応してサーフェス番号が交互（裏面，表面）に変化していますが，これはヘッドの移動量のむだをなくして，アクセス速度を向上する効果を持っています．

表5-4　クラスタ番号と物理アドレスの対応関係（N_{88}BASICの場合）

(1) 3.5"2HD, 5"2HD, 8"2Dの場合

クラスタ番号	サーフェス番号	トラック番号	セクタ番号
0	0	0	1～26
1	1	0	1～26
2	0	1	1～26
3	1	1	1～26
4	0	2	1～26
⋮			
152	0	76	1～26
153	1	76	1～26

(2)3.5"2DD，5"2DDの場合

クラスタ番号	サーフェス番号	トラック番号	セクタ番号
0	0	0	1～16
1	1	0	1～16
2	0	1	1～16
3	1	1	1～16
4	0	2	1～16
⋮			
158	0	79	1～16
159	1	79	1～16

*　裏面用ヘッドと表面用ヘッドは，フロッピーディスクをはさんで，常に向かい合った状態で移動している．

⑶ディレクトリ(DIR)

ディスク上のデータをファイルという概念に基づいて管理するためのテーブルをディレクトリと呼んでいます．1ファイル当りに割り当てられている管理制御情報は16バイトです．フロッピーディスク1枚で管理できるファイルの個数を**表5-5**に示します．全部のファイルの管理制御情報の集合がディレクトリです．

1つのファイル当りに割り当てられている16バイトの管理制御情報は**表5-6**に示すように，いくつかのフィールドからなっています．

表5-5
管理可能なファイル数

タイプ	ファイル数
3.5"2DD 5"2DD	159
5"2HD/3.5"2HD 8"2D	151

表5-6　ディレクトリの内容(制御情報の詳細)

フィールド(バイト)	内　　容
0～5	ファイル名(ASCIIコードで記入)
6～8	拡張子
9	属性(注)
10	ファイルが格納されている先頭のクラスタ番号
11～15	未使用

(注)属性

ビットNo.	意　　味	
	ビット値＝0	ビット値＝1
b_0	非機械語	機械語形式
b_1～b_3	未使用	
b_4	書き込み可能	書き込み禁止
b_5	エディット可能	エディット禁止(Pオプション)
b_6	書き込み時のチェックなし	チェックあり
b_7	ASCII形式	非ASCII形式

⑷FAT(File Allocation Table)

FATは，個々のファイルが占有するディスク領域の管理を行うためのテーブルです．この場合の管理は，クラスタを最小単位として行います．各クラスタに1バイトの制御情報を割り当てていて，この制御情報をクラスタの個数分だけ集めた集合体がFATです．

FATの機能を理解するために，具体例に即して説明します．例えば，“×××”という名称のファイルがクラスタ番号35，37，30の順番に3つのクラスタを使用しているものとします．この場合，DIRの項ですでに述べたようにDIR上にはファイル名”×××”と先頭クラスタ番号35が記入されています．そしてFATには図5-2のような管理情報が書き込まれています．

クラスタ35，37には後続するクラスタが存在するので，FAT上の対応するフィールドには，後続のクラスタ番号が記入されています．最終クラスタ30に対応するフィールドには，すでに使用済みのセクタ数が書き込まれています．ただし，C0Hだけオフセットが加わった値となっていますが，これはクラスタ番号との混同を避けるためです．

なお，システムで使用されているクラスタに対応するフィールドには，FEHが記入されています．

図5-2　FATの内容

対応するクラスタ番号	→	2F	30	31	32	33	34	35	36	37	38
FATのフィールド（1バイト区切り）	→		C5					37		30	

(5)ディスクID

ディスクIDとは，そのディスクの属性を指定するための制御情報やそのディスクを識別するための情報が書き込まれている領域のことです．具体的には，**表5-7**のような構成になっています．

表5-7　ディスク IDの内容

フィールド（バイト）	サイズ（バイト）	内容
0	1	属性*1
1	1	ファイル数*2
2H～FFH	254	BASICテキスト*3

＊1

ビットNo.	意味	
	ビット値＝0	ビット値＝1
b_0～b_3	(常時0)	
b_4	書き込み可能	禁止
b_5	(常時0)	
b_6	書き込み時のチェックなし	チェックあり
b_7	(常時0)	

＊2　同時にオープンできるファイル数を指定（0H～FH）できる．指定したときには，オートスタートモードになり，

How many files (0-15) ?

は表示されず，自動的に指定値が採用される．通常は，FFHが記入されており，オートスタートモードではない．

＊3　オートスタート時に，最初に実行されるステートメントを書き込んでおくことができる．例えば，

RUN"×××"

の文字列に相当するASCIIコードの列を書き込んでおけば，"×××"という名前のプログラムが自動的に実行される．通常は，00Hか20Hの列が記入されている．

＊4　IDを書き込むには，システムディスクに含まれている「setinf.n88」を使用する．

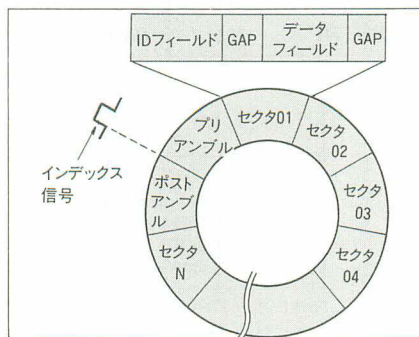
≡2.3≡ フォーマット

(1)セクタの構造

1トラック分の物理フォーマットは，**図5-3(1)**に示したように，通常，インデックスホールを検出したところから始まり，プリアンブルと呼ばれるもの，続いて，次々に指定したデータ長でデータ数だけのセクタを割り当てていき，最後に，次のインデックスホールまでの残りの余ったところにポストアンブルを書き込んでいきます．

図5-3(1)
フォーマット概略図

図5-3(2) フォーマット詳細

〔用語の説明〕
GAP：各セクタの最初や最後，そしてIDフィールドとデータフィールドなどの間にある余白部分
SYNC：IDフィールド・データフィールドの最初にある部分で，同期をとる(Synchronize)ためのデータ
IM(インデックスマーク)：トラックの最初であることを示すマーク
IDAM：(IDアドレスマーク)：IDフィールドの最初であることを示すマーク
CRC(Cyclic Redudancy Check)：ディスクから読み出されたデータにエラーが含まれていないかをチェックする
DAM(データアドレスマーク)：データフィールドの最初であることを示すマーク
DDAM(デリーテッドデータアドレスマーク)：DAMのかわりに，削除されたデータであることを示すマーク

すでに第5章2.1(2)で述べたように，物理アドレスが割り当てられている個々のセクタは，IDフィールドとデータフィールドからなっています．

そして，IDフィールドとデータフィールドの前後には，ディスクの回転数変動などの誤差からデータを保護するためにトラック上に設けられたGAPと呼ばれる領域があります．

IDフィールドは，次に続くデータフィールドを示す指標になる部分であり，**図5-3(2)**に示したように，各々のセクタのディスクのなかでの位置を示す論理アドレスやセクタの長さなどを示すC，H，R，Nという4バイトのID情報（ここでは，セクタIDと呼びます）が記入されています．

データフィールドは，実際にRead/Writeされるデータそのものが入っている領域で，N＝1ならば256バイトです．

(2)セクタシーケンス

物理アドレスでは，**図5-3(1)**で示したように，セクタ番号が円周に沿って連続して割り当てられています．

ディスク上のデータをアクセスするときには，セクタ単位で行われますが，同一のトラック上にある複数のセクタからデータをアクセスしようとした場合，ディスクのアクセスとアクセスの間には処理のための時間が必要なため，連続してアクセスのみ行うことはできません．そこで，処理が終わってアクセスを始めたら次のセクタがヘッドの位置に回転してくるように，各セクタに対して**図5-4**に示したように，セクタのアドレスを順番にせず，1つ飛ばしで物理アドレスとは異なる論理アドレスというものを割り当てています．

図5-4　物理アドレスと論理アドレスのセクシーケンスの違い

物理セクタ番号	01	02	03	04	05	06	07	08	09	～	17	18	19	1A
論理セクタ番号	01	0E	02	0F	03	10	04	11	05	～	0C	19	0D	1A

(注)数値は16進表現

3 DISK BIOS

3.1 DISK BIOSの概要

フロッピーディスク装置に対するデータのI/O*制御を行っているのが，μPD765AというLSIです．一般に，このLSIのことをFDC(フロッピーディスクコントローラ)と呼んでいます．FDCは，汎用性の高いコントローラなので，640KB/1MBのいずれの装置も制御できるようになっています．また，ディスク装置におけるデータのI/O制御に必要な種々の制御機能を持っています．FDCを動作させるためには，FDCに固有のDISK I/O制御命令を送出しなければなりません．DISK I/O制御命令は10数種類あります．いずれの制御命令も，その命令に1対1に対応するコマンドコードを専用I/Oポートを介して送出することによって実行されます．ただし，大部分の制御命令はコマンドコードに引き続いて多数のパラメータを送出する必要があります．いずれの命令も多数のパラメータを伴っていて，しかもこれらの命令を複合させて使用するとなると，ユーザにとって大きな負荷となってしまいます．そこで，PC-98にはユーザがFDCをより簡単な手続きで，目的とする機能で動作させることができるようにするためのソフトウェアが用意されています．それをDISK BIOS**と呼んでいます．DISK BIOSは10数種類のBIOSコマンドからなっています．**表5-8**に示すように，各BIOSコマンドには，1バイトのBIOSコマンドコードが割り当てられています．

なお，個々のBIOSコマンドについての詳細は，次項で述べます．

---

*　I/O＝Input/Output

**　BIOS＝Basic Input Output System

表5-8　DISK BIOSコマンドとDISK BIOSコマンドコード(1M／640Kドライブ用)

コマンド名	コマンドコード								機　　能
	b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0	
INITIALIZE	0	0	0	0	0	0	1	1	FDCのイニシャライズを行う．
FORMAT TRACK	0	MF	$\overline{r}$	SEEK	1	1	0	1	1トラック分のフォーマッティングを行う．
READ DATA	MT	MF	$\overline{r}$	SEEK	0	1	1	0	ディスクのデータを読み取る．
WRITE DATA	MT	MF	$\overline{r}$	SEEK	0	1	0	1	ディスクにデータを書き込む．
SEEK	0	0	$\overline{r}$	1	0	0	0	0	指定された位置へヘッドを移動させる．
RECALIBRATE	0	0	$\overline{r}$	0	0	1	1	1	基準位置(トラック0)へヘッドを移動させる．
VERIFY	MT	MF	$\overline{r}$	SEEK	0	0	0	1	READ DATAとほぼ同じ，ただしDMA転送はしない．
SENSE	0	0	0	0	0	1	0	0	デバイスの状態や属性を調べる．
READ ID	0	MF	$\overline{r}$	SEEK	1	0	1	0	トラック上のエラーのないIDを読み取る．
WRITE DELETED DATA	MT	MF	$\overline{r}$	SEEK	1	0	0	1	DDAM付きのデータを書き込む．
READ DELETED DATA	MT	MF	$\overline{r}$	SEEK	1	1	0	0	DDAM付きのデータを読み取る．
READ DIAGNOSTIC	0	MF	$\overline{r}$	SEEK	0	0	1	0	ID/DATA部からのエラーが検出されても読み取り続行する．

MT＝1：マルチトラック　　MF＝1：倍密度　　F＝1：リトライあり　　SEEK＝1：SEEKを伴う
　　0：シングルトラック　　　0：単密度　　　0：リトライなし　　　　0：SEEKなし

3.2 DISK BIOSの使用方法

DISK BIOSの使用方法，つまり，DISK BIOSコマンドの実行方法について，その手順を以下に説明します．1MB装置，640KB装置ともに共通要素が多いので，まとめて説明します．

①コマンドコードをレジスタAHにセットする（AH←コマンドコード）．

②必要ならば，所定のレジスタにパラメータをセットする．

③ソフトウェア割り込みの実行（INT 1BH*）．

個々のDISK BIOSコマンドについての詳細を以下に述べます．

各コマンドの説明は，下記の5つの項目で構成されています．

項目名	解説
〔機能〕	コマンドの機能説明
〔コマンドコード〕	上記手続きの①に対応
〔入力〕	上記手続きの②に対応
〔割り込みコード〕	上記手続きの③に対応
〔出力〕	コマンド実行後に，戻されるパラメータを列記している．

なお，BIOSコマンド実行後に，AHレジスタに戻されるステータス情報コードの一覧表を**表5-9**に示しておきます．

表5-9　ステータス情報とステータスコード

ステータスコード	ステータス呼称	説　　明
00H	Ready(SENSEコマンド時) Normal end(上記以外)	
10H	Write protect	媒体は存在するが，ライトプロテクトされている
20H	DMA Boundary	メモリがバンクにまたがる／番地が奇数で指定されている
30H	End of cylinder	データ長が1回の転送容量を越えている
40H	Equipment check	デバイス異常
50H	Over run	メモリからセクタへ，時間内にデータ転送できない
60H	Not ready	ユニットがノットレディ状態
70H	Not writable	WRITE PROTECT信号がON
80H	Error	
90H	Time out	
A0H	Data error(IDコマンド)	IDの読み出しでCRCエラー発生
B0H	Data error(DATAコマンド)	
C0H	No data	指定のセクタがない
D0H	Bad cylindar	
E0H	Missing address mark(IDコマンド)	トラック内に指定セクタがなく，IDもない
F0H	Missing addrees mark(DATAコマンド)	
01H		両面媒体がセットされている

＊　DISK BIOS用の割り込みベクタコードとして，1BHが割り当てられている．

3.3 DISK BIOSコマンド

(1)INITIALIZEコマンド

[機能]

デバイスアドレスで指定するディスク装置に関して初期化を行う．

①FDCの初期化

②システム共通エリアの初期化

③FDCに対して動作モードを設定

④各ユニットにRECALIBRATEコマンドを実行

[割り込みコード]

INT　1BH

[コマンドコード]

AH←03H

[入力]

AL←DA/UA（デバイスアドレス）

注）上位4バイト＝デバイス番号
　　下位4バイト＝ユニット番号

[出力]

(3)READ DATAコマンドと同じ

システム共通エリアのうち，下記ブロックが初期化される（表2-2(1)参照）

相対アドレス	サイズ（バイト）	フィールド名	説明
15CH	2	DISK_EQIP	ディスク装置の接続状況を示す情報
15EH	2	DISK_INT	ディスク装置からの割り込みフラグ
164H	32	DISK_RSLT	FDCから戻される制御情報

⑵FORMAT TRACKコマンド

[機能]

指定したデバイスの指定したトラックに対してフォーマッティングを行う（図5-3参照）．

①1トラック分の指定されたセクタ長，トラックあたりのセクタ数，ギャップ長にしたがってフォーマットを書き込む．

②各セクタのセクタID部にセクタIDを書き込み，各セクタの論理アドレスを定義する．ただし，各セクタのセクタIDは，指定したバッファ上にあらかじめ列記してあるものとする．このバッファの内容を順次セクタID部に書き込む．

③各セクタのデータ部に，指定したデータパターンをセクタ長で指定した長さ分だけ繰り返し書き込む．

[割り込みコード]

INT　1BH

[コマンドコード]

[入力]

AL←DA／UA（デバイスアドレス）

CL←C（シリンダ番号）

DH←H（ヘッド番号）

CH←N（セクタ長指定コード）

DL←D（データ部に書き込むビットパターン）

ES←セクタIDの列を格納しておくバッファの先頭アドレス（セグメント）

BP←セクタIDの列を格納しておくバッファの先頭アドレス（オフセット）

BX←DTL（セクタIDの列の長さ：バイト）．通常DTL＝4×セクタ数

[出力]

⑶READ DATAコマンドと同じ

(3)READ DATAコマンド

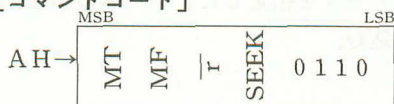
[機能]

指定したデバイスの指定した物理アドレスのセクタのデータ部から，指定した長さ分だけデータを読みだし，指定したバッファへ転送する．

[割り込みコード]

INT 1BH

[コマンドコード]

	MSB				LSB
AH→	MT	MF	$\overline{r}$	SEEK	0110

ビット名	解　　説	ビット値=0	ビット値=1
MT	マルチトラックの読み出し指定	シングルトラック	マルチトラック
MF	記録密度	単密度	倍密度
$\overline{r}$	エラー時のリトライ	有(8回)	無
SEEK	ヘッド移動	無(現在位置のまま)	SEEK

[入力]

AL←DA/UA（デバイスアドレス）

CL←C（シリンダ番号） ┐
DH←H（ヘッド番号） ├ 物理アドレス* ┐
DL←R（セクタ番号） ┘ ├ セクタID（図5-3参照）
CH←N（セクタ長指定コード） ┘

N	0	1	2	3
セクタ長(バイト)	128	256	512	1024

ES←データを格納するバッファの先頭アドレス（セグメントアドレス）

BP←データを格納するバッファの先頭アドレス（オフセットアドレス）

BX←DTL（データ長：バイト）

＊　1MBと640KBで異なる

	DA/UA	C	H	R
1MB	1 0 0 1 0 0 b_1 b_0	0〜76	0, 1	1〜26
640KB	0 1 1 1 0 0 b_1 b_0	0〜79	0, 1	1〜16

[出力]

CF← 0 (正常終了時)/ 1 (異常終了時)

AH←ステータス情報コード

システム共通エリアの値がセットされる. (1)INTIALIZEコマンド参照

(4)WRITE DATAコマンド

[機能]

指定したバッファ上の指定した長さのデータを指定したデバイスの指定した物理アドレスのセクタのデータ部へ書き込む.

[割り込みコード]

INT 1BH

[コマンドコード]

AH←	MT	MF	$\overline{r}$	SEEK	0 1 0 1

[入力]

(3)READ DATAコマンドと同じ

[出力]

(3)READ DATAコマンドと同じ

(5)SEEKコマンド

[機能]

指定したデバイスのヘッドを指定したシリンダ番号へ移動させる.

[割り込みコード]

INT 1BH

[コマンドコード]

AH←10H

[入力]

AL←DA/UA (デバイスアドレス)

CL←C (シリンダ番号)

[出力]

(3)READ DATAとコマンドと共通

⑹RECALIBRATEコマンド

［機能］

指定したデバイスのヘッドをシリンダ番号0（基準位置）（デバイスからTRACK 0信号を検出するまで）へシークさせる．

［割り込みコード］

INT　1BH

［コマンドコード］

AH←07H

［入力］

AL←DA/UA（デバイスアドレス）

［出力］

⑶READ DATAコマンドと共通

⑺VERIFYコマンド

［機能］

指定したデバイスの指定した物理アドレスのセクタのデータ部から指定した長さ分だけデータを読み出す．ただし，バッファへのデータ転送はしない．

［割り込みコード］

INT　1BH

［コマンドコード］

AH← | MT | MF | $\overline{T}$ | SEEK | 0 0 0 1 |

［入力］

⑶READ DATAコマンドと共通

［出力］

⑶READ DATAコマンドと共通

ただし，デリーテッドデータアドレスマーク(DDAM)を検出してもそのセクタをスキップして処理を続行する．

(8)SENSEコマンド

［機能］

指定したデバイスの状態を調べる．

①媒体の存在の有無

②プロテクトの有無

③媒体種別

［割り込みコード］

INT　1BH

［コマンドコード］

AH←04H

［入力］

AL←DA/UA（デバイスアドレス）

［出力］

CF←00H（正常時）/01H（異常時）

AH←ステータス情報コード

コード	略　称	解　　説
10H	WP	媒体がライトプロテクト状態．
60H	NR	媒体がセットされていない．

注）ただし，LSB(b_0)＝ { 0：片面媒体 / 1：両面媒体 }

⑼READ IDコマンド

［機能］

指定されたデバイスの指定したサーフェスの指定されたトラック上の最近接のセクタからセクタIDを読み出す．

［割り込みコード］

INT 1BH

［コマンドコード］

AH←	0	MF	$\overline{T}$	SEEK	1010

［入力］

AL←DA/UA（デバイスアドレス）

CL←C（シリンダ番号）

DH←H（ヘッド番号）

［出力］

⑶READ DATAと同じ．

ただし，下記のものも加えて出力される．

CL←C（論理シリンダ番号）
DH←H（論理ヘッド番号）
DL←R（論理セクタ番号）
（以上3項目：論理アドレス）
CH←N（セクタ長指定コード）
（以上4項目：セクタID）

⑽WRITE DELETED DATAコマンド

［機能］

デリーテッドデータアドレス(DDAM)マーク付きのデータを書き込む．

［割り込みコード］

INT 1BH

［コマンドコード］

AH←	MT	MF	$\overline{T}$	SEEK	1001

［入力］

⑷WRITE DATAと同じ

［出力］

⑷WRITE DATAと同じ

(11)READ DELETED DATAコマンド

[機能]

デリーテッドデータアドレス(DDAM)マーク付きのデータを読み込む.

[割り込みコード]

INT　1BH

[コマンドコード]

AH←

MT	MF	$\overline{\text{T}}$	SEEK	1 1 0 0

[入力]

(3)READ DATAと同じ

[出力]

(3)READ DATAと同じ

(12)READ DIAGNOSTICコマンド

[機能]

ID/データのエラーが検出されても読み取りを続行する.

[割り込みコード]

INT　1BH

[コマンドコード]

AH←

0	MF	$\overline{\text{T}}$	SEEK	0 0 1 0

[入力]

(3)READ DATAと同じ

[出力]

(3)READ DATAと同じ

4 DISK LIO

4.1 DISK LIOの概要

PC-98には,フロッピーディスク装置におけるデータのI/O制御を行うソフトウェアとしてDISK BIOSが用意されていることは，すでに前節で述べました.

DISK BIOSは，ハードウェアに密着した制御ソフトであるので，キメの細かい動作を指定できる反面，複合化してより複雑な動作をさせようとする場合には，手続きが複雑になってしまいます.

このデメリットを解消するために用意されているのが,DISK LIOと呼ばれるソフトです.DISK LIOは系統化されたいくつかのコマンドとしてユーザに用意されています.ディスク装置のI/Oに必要な一連の動作をDISK BIOSコマンドを組み合わせて表現したものが,DISK LIOの１つのコマンドとして提供されていると考えてよいと思います.

DISK LIOコマンドは,全部で９種類*あり,それぞれのコマンドにはDISK LIOコマンドが割り当てられています.

DISK LIOコマンドの一覧を**表5-10**に示します.

なお，DISK LIOはN88BASICでのみ有効となります.

表5-10 DISK LIO コマンド

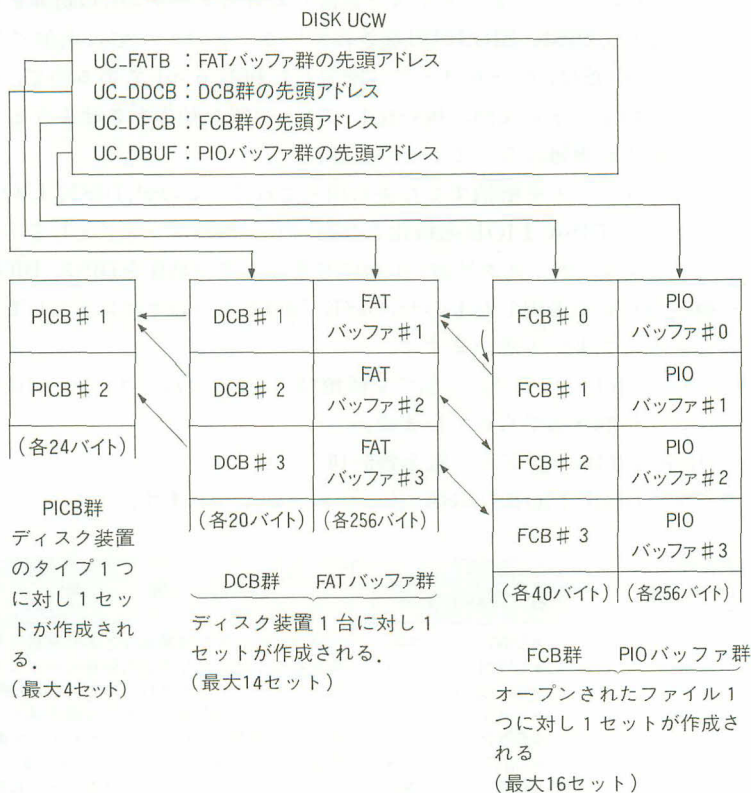
コマンド名称	コマンドコード	備考
*DINT	B4H	DISK LIOに必要な作業域などの初期設定をする
*OPEN	01H	指定したファイルを指定の処理モードでオープンする
*CLOSE	02H	指定したファイルをクローズし，FATを更新する
*SGET	03H	ディレクトリの内容をメモリに読み込む
*SREP	04H	*SGETの逆動作(メモリからディスクへ書き戻す)
*SDEL	05H	指定したファイルをディレクトリのエントリから削除する
*GET	06H	指定したファイルからデータをメモリに読み込む
*PUT	07H	*GETの逆動作(メモリからディスクへ転送する)
*PIO	08H	指定したデバイス上の特定アドレスに対して入出力動作を実行する
*SENS	09H	指定したデバイスの状態と媒体種別を調査する

＊ 本書では，FDDに関するものだけ扱っていて，HDに関するものは除外している.

4.2 DISK LIOの制御関連図

DISK LIOの制御関連図を図5-5に示します．

DISK LIOには，制御領域としてPICB，DCB，FCB*の3種類が，バッファとしてFATバッファ，PIOバッファの2種類が用意されています．これらの領域の先頭アドレスは，DISK UCW**に格納されています．

図5-5　DISK LIO制御関連図

*　PICB＝Physical I/O Control Block. 表2-4参照
DCB＝Device Control Block. 表2-5参照
FCB＝File Control Block. 表2-6参照

**　UCW＝Unit Control Work. 表2-3参照

図5-5の解説を以下に述べます．

まず，ディスク装置のタイプ(種類)の数に相当するだけのPICB#n_1が作成されています．PICB#n_1には，そのタイプに固有のI/O制御情報が格納されます．

次に，ディスク装置の台数に相当するだけのDCB#n_2とFATバッファ#nが作成されます．DCB#n_2には，そのディスク装置の動作状態やファイル管理情報（FAT，DIR）のポインタアドレスなどが格納されています．ディスク装置がI/O動作をするときには，自分の属するタイプに対応したPICB#nを参照し，その制御情報のしたがうことになります．

また，DISK BASIC起動時に設定する”同時オープン可能ファイル数”に相当するだけのFCB#n_3とPIOバッファ#nが作成されます．

FCB#n_3には，ファイル番号#n_3に割り付けられているファイル名やそのファイルへのアクセス状態，あるいはファイル番号#n_3が割り付けられているディスク装置に対応するDCB#n_2のポインタアドレスなどが格納されています．

図5-5を見ると，装置#1，#2は同じタイプであり，装置#1には現在，ファイル番号#0，#1の２つがオープンされていることがわかります．

4.3 DISK LIOの使用方法

DISK LIOの使用方法，つまり，DISK LIOコマンドの実行方法について，その手順を説明します．

①コマンドコードをレジスタAHにセットする（AH←コマンドコード）

②必要ならば，所定のレジスタにパラメータをセットする．制御領域（UCW，DCB，FCB，PICB）の各フィールドにパラメータをセットする．

③ソフトウェア割り込みの実行

INT　0B0H

個々のDISK LIOコマンドの詳細については，順を追って以下に説明します．各コマンドの説明は，下記の５つの項目で構成されています．

項目名	解説
〔機能〕	コマンドの機能解説
〔コマンドコード〕	上記手続きの①に対応
〔入力〕	上記手続きの②に対応
〔割り込みコード〕	上記手続きの③に対応
〔出力〕	コマンド実行後に戻されるパラメータを配列している

なお，DISK LIO使用に先立って，まずDISK LIO初期化コマンドである＊DINTを実行しておく必要があります．

また，各コマンド実行後に，レジスタAHにリターンコードが戻されますが，このリターンコードの意味する内容は，表5-11に示す通りです．

表5-11　リターンコード一覧表

リターンコード	内　容	備　考
69	Bad allocation table	正しくFATを読み込めなかった
70	Bad drive number	ドライブ番号不正のチェック
56	Bad file name	ファイル名不正のチェック
52	Bad file number	ファイル番号不正のチェック
71	Bad track/sector	DCFの値に誤りがある
72	Deleted record	
67	Disk already mounted	
68	Disk full	ディスク上の空きエリアがない
63	Disk not mounted	
64	Disk I/O error	入出力エラーが発生
62	Disk offline	媒体がセットされていない
65	File already exists	すでにファイルが存在している
54	File already open	ファイルがすでにオープンされている
53	File not found	ファイルが見つからない
60	File not open	
61	File write protected	ファイル属性により書き込み禁止
55	Input past end	ディレクトリを読み込みエラー検出
73	Rename across disks	
58	Sequential after PUT	
59	Sequential I/O Only	

≡4.4≡ DISK LIOコマンド

(1) *DINTコマンド

[機能]

DISK LIO使用に先立って，下記の制御領域のイニシャライズを行う．
ただし，制御領域のセグメントアドレスは0060Hとする．

①UCWの初期設定

②DCB，FCB，PICBの初期設定

③バッファ領域の確保

[割り込みコード]

INT　0B4H

[入力]

DS←制御領域先頭アドレス（セグメントアドレス）＝0060H
ES←制御領域先頭アドレス（セグメントアドレス）＝0060H
SS←制御領域先頭アドレス（セグメントアドレス）＝0060H
BP←制御領域先頭アドレス（オフセットアドレス）＝0000H

設定しておくべきUCWのフィールド（表2-3参照）

フィールド名称	相対アドレス	サイズ（バイト）	説明
UC_DOPN	504H	1	同時にオープンできるファイル数
UC_SRVT	505H	1	OS種別 01H：N_{88}-BASIC 02H：N-BASIC，5"FD 1D 03H：N-BASIC，5"FD 2D

[出力]

BP←制御領域最終アドレス（オフセットアドレス）

⑵ *OPENコマンド

［機能］

指定ファイルを指定デバイスにオープンする．ディスクID，FATを読み込み，マウント処理を行う．また，ファイルに対する処理モードを指定する．

［割り込みコード］

INT　0B0H

［コマンドコード］

AH←01H

［入力］

AL←DA/UA（デバイスアドレス）

BX←FCB#nの先頭アドレス

設定しておくべきFCBのフィールド（表2-6参照）

フィールド名称	相対アドレス	サイズ（バイト）	説明
FC_FID	06H	6	ファイル名
FC_EID	0CH	3	ファイル拡張子
FC_OPNM	01H	1	処理モード 80H：INPUTモード 40H：OUTPUTモード 41H：APPENDモード C0H：モード指定なし
FC_ATTR	0FH	1	ファイル属性

［出力］

AH←リターンコード

⑶ *CLOSEコマンド

［機能］

指定ファイルをクローズする．FATが更新されていれば，書き直す．

［割り込みコード］

INT　0B0H

［コマンドコード］

AH←02H

［入力］

BX←FCB#nの先頭アドレス

［出力］

AH←リターンコード

(4) *SGETコマンド

[機能]

ディレクトリ（DIR）をブロック単位（256バイト）で読み込む．個々のファイルに対応するFCBを作成する．

[割り込みコード]

INT　0B0H

[コマンドコード]

AH←03H

[入力]

AL←DA/UA（デバイスアドレス）

BX←FCB#0の先頭アドレス

設定しておくべきFCBのフィールド（**表2-6参照**）

フィールド名称	相対アドレス	サイズ（バイト）	説明
FC_FID	06H	6	ファイル名
FC_EID	0CH	3	ファイル拡張子

ただしFC.FIDの先頭1バイト＝ 00H……NEXT処理 / FFH……FIRST処理 / 以外……RANDOM処理

[出力]

AH←リターンコード

BX←FCB#nの先頭アドレス**

当コマンド実行後に設定されるFCBのフィールド（**表2-6参照**）

フィールド名称	相対アドレス	サイズ（バイト）	説明
FC_FID	06H	6	ファイル名
FC_EID	0CH	3	ファイル拡張子
FC_ATTR	0FH	1	ファイル属性
FC_FCLS	10H	2	先頭クラスタ番号
FC_EOD	15H	3	最終レコードアドレス
FC_LRNO	18H	2	最終レコード番号

*　FCB＃0の先頭アドレスはUCW上のフィールドUC-DFCBに格納されている．したがって，FCB＃nの先頭アドレスは，FCB＃0の先頭アドレス＋40×nで与えられる（図5-5参照）．ただし，セグメントアドレス＝0060H

(5) ＊SREPコマンド

[機能]

＊SGETコマンドで読み込まれたディレクトリブロックの内容をディスクのディレクトリ領域に書き戻す.

[割り込みコード]

INT　0B0H

[コマンドコード]

AH←04H

[入力]

AL←DA/UA（デバイスアドレス）

BX←FCB#0の先頭アドレス

設定しておくべきFCBのフィールド（表2-6参照）

フィールド名称	相対アドレス	サイズ(バイト)	説明
FC_FID	06H	6	ファイル名
FC_EID	0CH	3	ファイル拡張子
FC_ATTR	0FH	1	ファイル属性

[出力]

AH←リターンコード

⑹＊SDELコマンド

［機能］

指定ファイルを削除する．ディレクトリ上より，指定ファイルのエントリを削除し，ファイル領域の開設およびFATの更新を行う．

［割り込みコード］

INT　0B0H

［コマンドコード］

AH←05H

［入力］

AL←DA/UA（デバイスアドレス）

BX←FCB#0の先頭アドレス

設定しておくべきFCBのフィールド（表2-6参照）

フィールド名称	相対アドレス	サイズ（バイト）	説明
FC_FID	06H	6	ファイル名
FC_EID	0CH	3	ファイル拡張子

［出力］

AH←リターンコード

⑺＊GETコマンド

[機能]

指定したファイルのデータをブロック単位（256バイト）で読み込む.

[割り込みコード]

INT　0B0H

[コマンドコード]

AH←06H

[入力]

BX←FCB#nの先頭アドレス

設定しておくべきFCBのフィールド（**表2-6参照**）

フィールド名称	相対アドレス	サイズ(バイト)	説　　明
FC_NRNO	1DH	2	レコード番号

[出力]

AH←リターンコード

⑻＊PUTコマンド

[機能]

指定したファイルのデータをブロック単位(256バイト)で書き込む. 事前にOPENコマンドより指定ファイルがオープンされていなければならない.

[割り込みコード]

INT　0B0H

[コマンドコード]

AH←07H

[入力]

BX←FCB#nの先頭アドレス

設定しておくべきFCBのフィールド（**表2-6参照**）

フィールド名称	相対アドレス	サイズ(バイト)	説　　明
FC_NRNO	1DH	2	レコード番号

[出力]

AH←リターンコード

⑼＊PIOコマンド

［機能］

指定したデバイスの特定アドレスに対して入出力を行う.

［割り込みコード］

INT　0B0H

［コマンドコード］

AH←08H

［入力］

AL←DA/UA（デバイスアドレス）

設定しておくべきPICBのフィールド（表2-4参照）

フィールド名称	相対アドレス	サイズ（バイト）	説明
PI_CMD	01H	1	DISK BIOSコマンド
PI_DTS	04H	2	データセグメントアドレス
PI_DTA	02H	2	データアドレス
PI_DTL	06H	2	データ長
PI_DCF	08H	6	セクタの物理アドレス

［出力］

AH←リターンコード

⑽＊SENSコマンド

［機能］

指定したデバイスの状態および媒体の識別を行う．

［割り込みコード］

INT　0B0H

［コマンドコード］

AH←09H

［入力］

AL←DA/UA（デバイスアドレス）

［出力］

AH←リターンコード

当コマンド実行後に設定されるDCBのフィールド（表2-5参照）

DC-DSTS（ビットD_2）	媒体種別

第6章

CONTENT

1 概要

本章では，PC-98に標準あるいはオプションで用意されているインターフェースについて解説します．ここで取り扱うインターフェースの種類は**表6-1**の通りです．

以下では，それぞれのインターフェースの特長を述べるとともに，PC-98にはそれぞれのインターフェースを介して，他の周辺機器や端末などとの間でデータの入出力を行うための基本ソフトウェア（それぞれの名称は**表6-1**を参照）が用意されているので，その使用法についても説明します．

表6-1 各インターフェースの種類

インターフェースの名称	基本入出力ソフトウェアの名称
RS-232C　インターフェース	RS-232C　BIOS
GP-IB　インターフェース	GP-IB　BIOS
マウス　インターフェース	マウス　BIOS
プリンタ　インターフェース	プリンタ　BIOS

2 RS-232Cインターフェース

2.1 RS-232Cインターフェース概要

PC-98はRS-232Cインターフェースを標準実装しています．したがって，RS-232Cインターフェースを備えている他の機器と接続して，シリアルデータ通信を行うことができます．

PC-98のROM上には，RS-232Cインターフェースを介してシリアルデータ通信を行うための制御プログラム（RS-232C　BIOSと呼ぶ）がすでに用意されているので，ユーザは，このBIOSを利用することにより，シリアルデータ通信を容易に実現することができます．

以降では，まずRS-232Cインターフェース規格についての一般的な解説を行い，次にRS-232C　BIOSの機能と使用法について説明します．

2.2 RS-232Cインターフェース規格

RS-232Cは，本来は**図6-1**に示すように，端末機器とモデム*の間のインターフェースを標準化**することを目的として定めたシリアルデータ通信に関するインターフェース規格です．しかし，最近はパーソナルコンピュータの著しい普及にともなって，パーソナルコンピュータ間の通信やパーソナルコンピュータと周辺機器（プリンタ，X-Yプロッタ，デジタイザetc）との接続にも多用されるようになりました．RS-232Cインターフェースの使用形態の例を**図6-2**に示します．

＊　通常のデジタル・シリアル信号を電話回線に乗せられる信号に変調する機能と，電話回線を介して送られてきた変調信号を元のデジタル・シリアル信号に復調する機能を持っている．

＊＊信号名，信号の電圧値などに関する規定のみ．プロトコルなどについては規定していない（自由度が広い）．

図6-1 本来のRS-232Cの使用形態

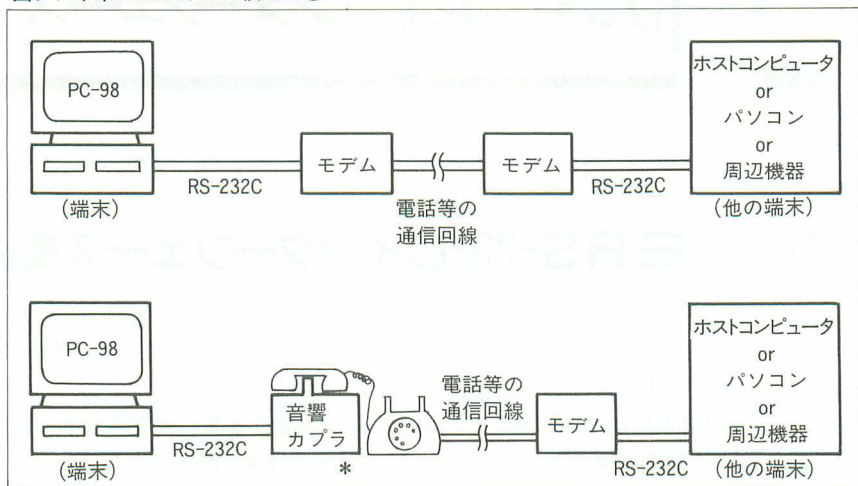
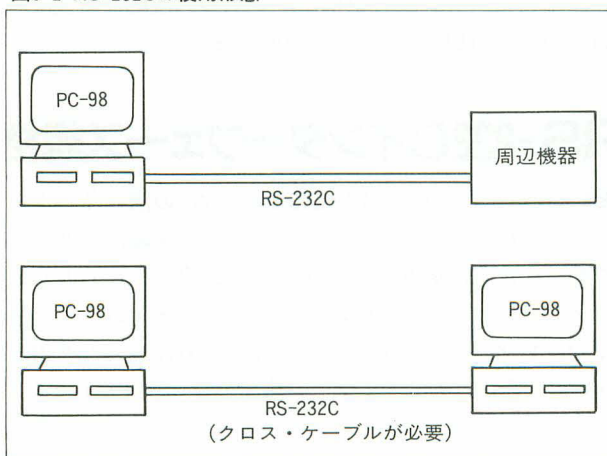
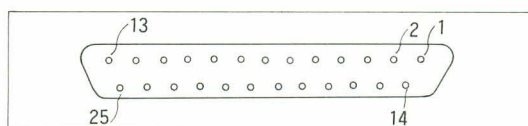



図6-2 RS-232Cの使用形態

* 音響カプラの機能は，本質的にはモデムの機能と同じであるといえる．ただ，変調された信号を音声化して，電話機との間で送受する点だけが異なる．

(1)RS-232Cのコネクタの形状と信号

RS-232Cのコネクタの形状と信号の説明を**図6-3**に示します．なお，すでに述べたように，RS-232Cは元来，端末とモデムの間のインターフェースを標準化したものであるということを考慮したうえで，信号の意味を理解して下さい．

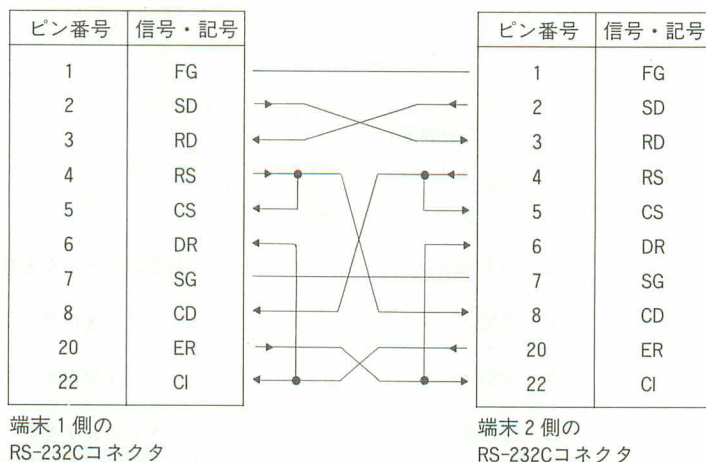
図6-3 RS-232Cのコネクタの形状と信号の説明

RS-232Cコネクタの形状とピン番号配置
(PC-98裏面)

ピン番号	信号		信号の向き 端末↔モデム	説明
	記号	名称		
1	FG	保安用グランド	――――	機器のシャーシを接続
2	SD	送信データ	――→	シリアル送信データ
3	RD	受信データ	←――	シリアル受信データ
4	RS	送信要求	――→	モデムに対するデータ送信要求
5	CS	送信可	←――	モデムからの送信可能の応答(端末からの送信要求に対する応答)
6	DR	データセットレディ	←――	モデムが送受信動作可能であることを，端末に伝える
7	SG	信号用グランド	――――	信号の基準電位
8	CD	受信キャリア検出	←――	相手側端末が送信状態にあることを伝える
20	ER	端末レディ	――→	端末が送受信動作可能であることを，モデムに伝える
22	CI	被呼表示	←――	相手側端末から呼び出しが生じていることを伝える

⑵接続方法

端末とモデムの間をRS-232Cで接続する場合には，特別注意すべき点はありません．以下では，モデムを介さず，端末間あるいは端末と周辺機器間を直接RS-232Cインターフェースで接続する場合の注意点を述べます．

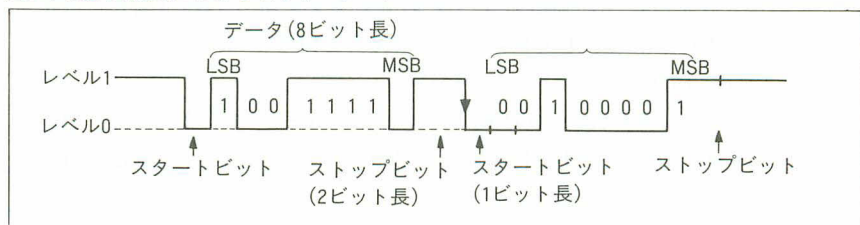
直接接続する場合には，ケーブルを**図6-4**に示すように，たすき掛け状に接続する必要があります．

図6-4 RS-232インターフェース間を直結する場合の結線方法（モデムを介さない場合）

2.3 調歩同期式

シリアル伝送方式は，大きく分けて同期式と非同期式(調歩同期式とも言う)に分けられます．前者は高速通信が望める反面，ハードウェアが複雑になるというデメリットがあるため，パソコンでRS-232Cを用いたシリアル伝送を行う場合には，多くの場合，非同期式が採用されます．

以下では，非同期式，つまり調歩同期式について説明します．

8ビットデータを送信する場合を例にとります．調歩同期式では，データの前後にスタートビット（レベル0）とストップビット（レベル1）を加えて，データの始まりと終わりを知らせています．通信時のビットパターンを**図6-5**に示します．

図6-5 調歩同期式におけるビットパターン

① データを送信していないときは，常にレベル1の状態にあります．この状態をアイドル状態と呼ぶ．

② スタートビットの立ち下がりのタイミングを基準にして，受信側は受信開始をする．最終のストップビットを検出することで，受信が正常に実行されたことを確認する．再び，スタートビットの立ち下がりを検出して，次のデータの受信を行う．

2.4 RS-232C BIOS

RS-232C BIOSとは，RS-232Cインターフェースを介して，シリアルデータ通信を行うためのソフトウェアであり，すでにPC-98のROM上に用意されています．元来，信号の入出力制御は繁雑なものですが，BIOSでは入出力制御に必要な処理内容をいくつかのBIOSコマンドとして系統化しているので，ユーザも容易に利用することができます．

RS-232C BIOSを利用する場合の手続きを以下に説明します．

① 実行したいRS-232C BIOSコマンドに対応するコマンドコードをレジスタAHにセットする．
コマンドの種類によっては，何種類かのレジスタの値をセットしておく必要がある．

② レジスタ設定後，下記命令によってソフトウェア割り込み（割り込み番号は19H）を実行すれば，BIOSコマンドが実行される．

INT 19H

この手続きは，高級言語におけるサブルーチンコールの手続きによく似ています．サブルーチンコールする際に引数を指定しますが，これはBIOSにおけるレジスタ設定に対応しています．

以下では，個々のBIOSについて説明します．なお，[入力]の項には，BIOSコマンド実行に先立ち設定しておくべきレジスタを列記しています．[出力]の項には，BIOSコマンド実行によって書き換えられるレジスタを列記しています．

(1)初期化コマンド

[機能]

RS-232Cインターフェースの初期設定を行います．

①μPD8251A*のモード設定

②タイマμPD8253Cのカウンタ２の設定(カウンタ２でデータの伝送速度を規定している)．

---

* μPD8251Aは，シリアル通信汎用インターフェース機能を持つLSI
インテル社i8251Aと同等の機能を持つ．

③受信バッファの設定
④割り込み受け付け状態の設定

[割り込みコード]

INT　19H

[コマンドコード]

AH←00H

[入力]

AL←通信レートパラメータ

通信レートパラメータ	00H	01H	02H	03H	04H	05H	06H	07H
伝送速度（BPS）	75	150	300	600	1200	2400	4800	9600

ALに08H以上の値を設定した場合は1200bpsに設定される．

CH←μPD8251Aのモード設定値
(非同期モード)

MSB							LSB
S2	S1	EP	PEN	L2	L1	B2	B1

B2,B1 = =	ボーレートの設定 10　：×16 11　：×64
L2,L1 = = = =	キャラクタ長の設定(データのビット長) 00　：5ビット 01　：6ビット 10　：7ビット 11　：8ビット
PEN =	パリティ・イネーブル* 1　：ON（パリティチェックする），0：OFF
EP =	パリティ指定* 1　：奇数　，0：偶数
S2,S1 = = =	ストップビット長 01　：1ビット 10　：1.5ビット 11　：2ビット

＊ データビットとパリティビット（1ビット長）の中に含まれる“1”の値の個数の偶奇に着目してデータが正しく受信されたかどうかチェックします．例えば，パリティ指定を偶数にした場合には，“1”の総数が偶数になるようパリティビットへ“0”または“1”をセットして送信されます．受信側では“1”の総数が偶数であるか否かをチェックします．

CL←μPD8251Aに対するコマンド

MSB							LSB
X	IR	RTS	ER	SBRK	RXE	DTR	TXEN

記号	名　　称	設定値と機能	
TXEN	送信イネーブル	1＝イネーブル	0＝ディスエーブル
DTR	データターミナル・レディ	1＝ON	0＝OFF
RXE	受信イネーブル	1＝イネーブル	0＝ディスエーブル
SBRK	センド・ブレイク・キャラクタ	1＝07Hを送信	0＝07Hを送信しない
ER	エラーリセット	1＝エラーフラグPE,OE,FEをリセットする	0＝リセットしない
RTS	送信要求	1＝ON	0＝OFF
IR	内部リセット	1＝リセット*	0＝リセットしない

ES←受信バッファの先頭アドレス（セグメントアドレス）

DI←受信バッファの先頭アドレス（オフセットアドレス）

DX←受信バッファのサイズ（単位：バイト）

注）1バイトの通信データと1バイトのステータスデータがペアで格納される．したがって，通信データNバイトに対してバッファは2×Nバイト必要．通信データは，アドレスES：DI＋14から，1バイトおきに存在する．

BH←送信時のタイムアウト時間（TXRDYステータスの待ち時間）

注）上記数値×500msが実際の待ち時間

デフォルト値は02H（1秒）

BL←受信時のタイムアウト時間（RXRDY割り込みの待ち時間）

注）上記数値×500msが実際の待ち時間

デフォルト値は1EH（15秒）

［出力］

AH←リターンコード（正常終了時には00H）

---

* μPD8251Aをモードインストラクションフォーマットに戻す．

(2)受信データ長調査コマンド

[機能]

受信バッファ内のデータの長さを調べる．

[割り込みコード]

INT　19H

[コマンドコード]

AH←02H

[出力]

AH←リターンコード

- 00H：正常終了
- 01H：RS-232Cの初期設定がされていない．
- 02H：受信バッファがオーバーフロー

CX←受信データ長（単位：ワード）

注）データ長とステータスの2バイト（1ワード）が1組になっている．

(3)データ送信コマンド

[機能]

データを1バイト送信する．

[割り込みコード]

INT　19H

[コマンドコード]

AH←03H

[入力]

AL←送信データ（1バイト）

[出力]

AH←リターンコード

- 00H：正常終了
- 01H：RS-232Cの初期設定がされていない．
- 02H：受信割り込み処理において，受信バッファがオーバーフロー
- 03H：送受信時にμPD8251Aからの送受信可のステータス（TXRDY, RXRDY)*を受け取れなかった．

---

* (6)ステータス読み取りコマンドを参照.

(4)データ受信コマンド

[機能]

受信バッファ内のデータを読み出す.

[割り込みコード]

INT　19H

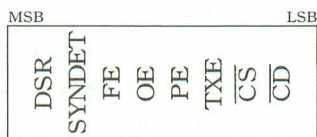
[コマンドコード]

AH←04H

[出力]

AH←リターンコード

CH←受信データ

CL←データ受信時のステータス情報

(5)μPD8251Aへコマンドを送信するコマンド

[機能]

μPD8251Aに対し，コマンドを出力する.

[割り込みコード]

INT　19H

[コマンドコード]

AH←05H

[入力]

AL←μPD8251Aに対するコマンド

MSB							LSB
×IR	RTS	ER	SBRK	RXE	DTR	TXEN	

注）各レジスタの意味については，(1)初期化コマンドにおけるレジスタCLに関する部分を参照.

[出力]

AH←リターンコード

- 00H：正常終了
- 01H：RS-232Cの初期化コマンドが未実行
- 02H：受信バッファのオーバーフロー

(6)ステータス読み取りコマンド

[機能]

μPD8251Aとシステムポートのステータスを読み取る．

[割り込みコード]

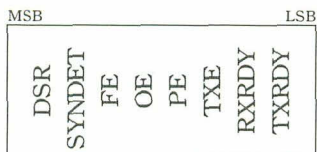
INT　19H

[コマンドコード]

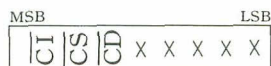
AH←06H

[出力]

AH←リターンコード

CH←μPD8251Aのステータス情報

記号	名　　称	設定値と機能	
DSR	データセットレディ	1＝ON	0＝OFF
SYNDET	ブレーク状態検出	1＝あり	0＝なし
FE	フレミングエラー	1＝発生	0＝なし
OE	オーバーランエラー	1＝発生	0＝なし
PE	パリティエラー	1＝発生	0＝なし
TXE	送信バッファ状態	1＝空	0＝満
RXRDY	受信状態	1＝レディ	0＝ビジー
TXRDY	送信状態	1＝レディ	0＝ビジー

CL←システムポートのステータス情報

MSB							LSB
$\overline{CI}$	$\overline{CS}$	$\overline{CD}$	X	X	X	X	X

記号	名　　称	設定値と機能	
$\overline{CI}$	着呼	1＝なし	0＝あり
$\overline{CS}$	送信	1＝不可	0＝可
$\overline{CD}$	受信キャリア検出	1＝なし	0＝あり

3 GP-IBインターフェース

最近の計測器機や各種周辺装置の中には，GP-IBインターフェースを備えたものが多く，これらを組み合わせることによって，高度な計測制御システムを容易に構成できるようになりました．

PC-98には，GP-IBインターフェースのハードウェアとして，GP-IBインターフェースボード（PC-9801-29）がオプションで用意されています．このボード上には，GP-IBインターフェースとしての機能を１つのLSIにまとめたμPD7210が実装されています．このLSIに制御命令を与えることにより，GP-IBインターフェースとしての様々な機能を引き出すことができます．また，このLSIを用いて，GP-IBインターフェースを介して，デジタル８ビットパラレルデータ通信を行うための基本制御ソフトウェアがすでにGP-IBインターフェースボード上のROM内に用意されていて，これをGP-IB BIOSと呼んでいます．ユーザは，このBIOSを利用することにより，GP-IBインターフェースを介したデータ通信を容易に実現することができます．

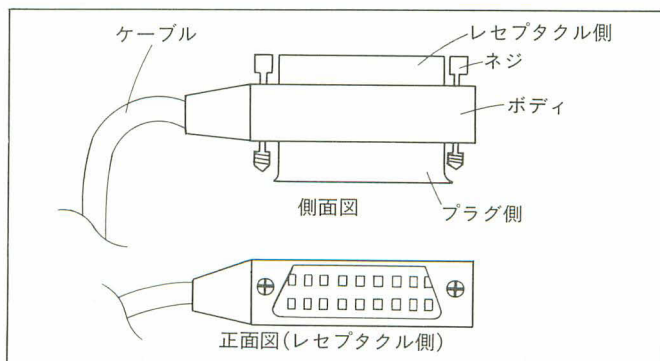
3.1 GP-IBインターフェースの概要

(1)GP-IBインターフェースの特長

GP-IB*は，標準化されたデジタル8ビットパラレルインターフェースバスであり，規格の正式名称は” IEEE Std 488-1975” と言います．

GP-IBでは，バス方式**を採用しているので，GP-IBインターフェースを備えた機器を複数台並列接続できます．したがって，GP-IBインターフェースを備えたコンピュータであれば，その1つのインターフェースにGP-IBインターフェースを備えた計測機器や周辺機器を複数台，同時に接続することができます．並列接続を効率よく行うために，GP-IB用ケーブル両端のコネクタは，図6-6に示すような形状になっています．コネクタは，プラグとリセプタクルが一体化されているので，複数のコネクタを重ね合わせて接続していくことができます．したがって，各機器はGP-IB用のコネクタ（レセプタクル側）を1つだけ備えていれば充分です．GP-IB機器間の接続方法としては，図6-7に示すような珠数型接続や星型接続も可能です．これに対し，GP-IBを採用する以前は，図6-8に示すような接続形態でした．各機器に専属のインターフェースをコンピュータ側も用意しておかなければなりませんでした．

以上よりわかるように，GP-IBのメリットは，”複数の機器からなる計測制御システムを容易に構成し，柔軟にレイアウトできる” 点にあります．

図6-6
GP-IBケーブル端末のコネクタ形状

*　General Purpose Interface Busの略称．

**同一の信号線を複数の機器間で共用する方式をバス（bus）方式という．

ところで，GP-IBでは，同一バスに複数台のGP-IB機器が並列接続されているので，互いを区別するために，GP-IBアドレスが割り当てられています．一般に，GP-IB上には，１台のコントローラ，１台のトーカ，１台以上のリスナが存在し，トーカからリスナへデータの送信が行われます．コントローラがGP-IB機器をトーカ，あるいはリスナに指定する動作を行っています．機器の指定は，GP-IBアドレスを用いて行います．

図6-7 GP-IB機器の接続形態

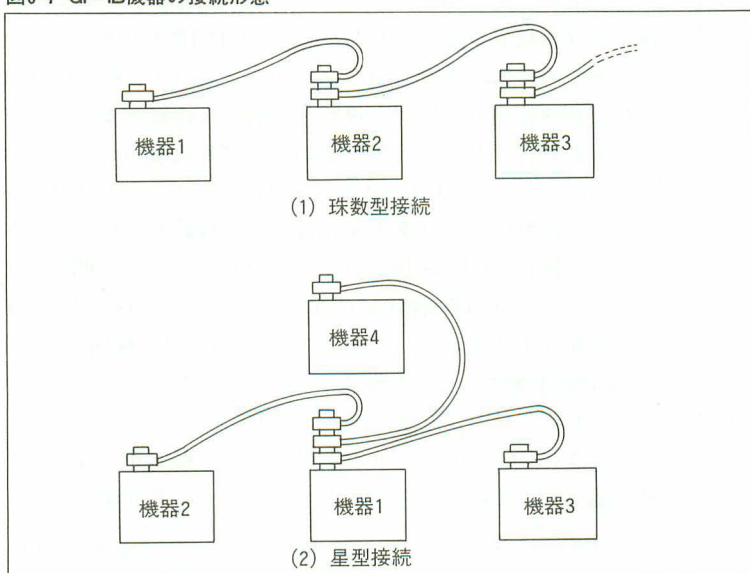
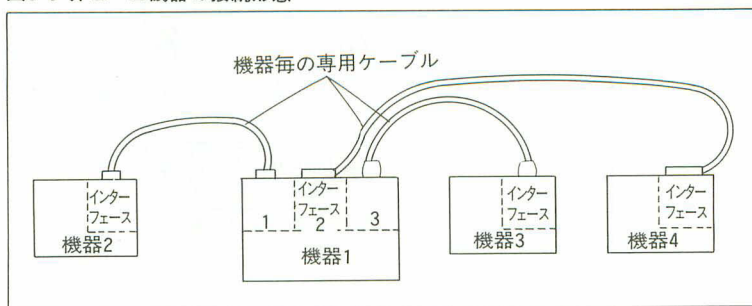
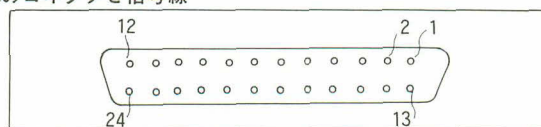



図6-8 非GP-IB機器の接続形態

(2)GP-IBの信号

GP-IBインターフェースのコネクタの形状と信号線についての説明を**図6-9**に示します.

信号線は，データバス，ハンドシェイクバス，管理バスの3種類に大別できます.

図6-9 GP-IBのコネクタと信号線

コネクタの形状と端子番号の配列(24ピン)

端子番号	名　称　説　明		端子番号	名　称　説　明	
1	DIO 1	データバス	13	DIO 5	データバス
2	DIO 2		14	DIO 6	
3	DIO 3		15	DIO 7	
4	DIO 4		16	DIO 8	
5	$\overline{\text{EOI}}$	管理バス	17	$\overline{\text{REN}}$	管理バス
6	$\overline{\text{DAV}}$	ハンドシェイクバス	18	DAV のグランド	
7	$\overline{\text{NRFD}}$		19	NRFDのグランド	
8	$\overline{\text{NDAC}}$		20	NDACのグランド	
9	$\overline{\text{IFC}}$	管理バス	21	IFC のグランド	
10	$\overline{\text{SRQ}}$		22	SRQ のグランド	
11	$\overline{\text{ATN}}$		23	ATN のグランド	
12	シールド		24	ロジック・グランド	

(i)データバス

データバス(DIO1〜8)は，8ビットのパラレルデータを機器相互間で送受するためのラインです．

データバスは，通常のデータを送るために使われるだけでなく，GP-IBに接続されている複数のGP-IBインターフェースの機器状態を設定するためのコマンド*を送るためにも使用されます．

データバス上の信号が通常のデータであるか，コマンドであるかの指標の役目を管理バスであるATNラインが果しています．

$\overline{\text{ATN}}$のレベル	モード	データ・バス上の信号
L	コマンドモード	コマンド
H	データモード	通常のデータ

(ii)ハンドシェイクバス

3本のハンドシェイクバスは，トーカとリスナの間のデータ通信の際に，両者のタイミングの制御を行うためのラインです．GP-IBでは，この3本のバスを用いて3線ハンドシェイクと呼ばれる方式で通信しています．各信号の意味と3線ハンドシェイクのタイムチャートを**図6-10**に示します．

(iii)管理バス

管理バスは，GP-IBインターフェース相互の状態を確認するためのラインです．5本の信号は，**表6-2**のようにコントローラが発するもの（$\overline{\text{ATN}}$，$\overline{\text{REN}}$，$\overline{\text{IFC}}$）とトーカが発するもの（$\overline{\text{SRQ}}$，$\overline{\text{EOI}}$）に分けられます．

---

* 具体的には，トーカ・アドレス，リスナ・アドレスなどがコントローラから送信され，GP—IBに接続されている複数のGP—IBインターフェースの中から，トーカ，リスナが決定される．トーカ，リスナが決定された後に，送信が開始できる．

図6-10 3線ハンドシェイク・信号の説明とタイムチャート

信号名	意味	意味
	Lレベル	Hレベル
$\overline{\text{DAV}}$ (data valid)	データバス上の信号が有効	データバス上の信号が無効
$\overline{\text{NRFD}}$ (not ready for data)	1つ以上のリスナがデータ受信不可能状態	すべてのリスナがデータ受信可能状態
$\overline{\text{NDAC}}$ (not data accepted)	1つ以上のリスナがデータ受信未完了	すべてのリスナがデータ受信完了

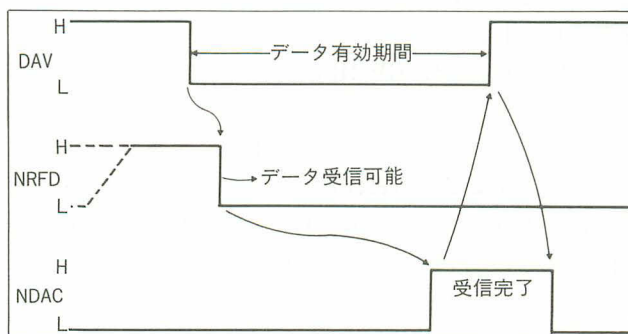



表6-2 管理バスの信号の説明

信号名	意味	
	Lレベル	Hレベル
$\overline{\text{ATN}}$ (attention)	データ・バスの信号が通常のデータであることを示す(データモード)	データ・バス上の信号がコマンドであることを示す(コマンド・モード)
$\overline{\text{REN}}$ (remote enable)	指定するGPIBアドレスの機器をリモート状態に設定する	〃 ローカル(マニュアル)状態に設定する
$\overline{\text{IFC}}$ (interface clear)	100μsec以上の"L"パルスでインターフェースをリセットする	
$\overline{\text{SRQ}}$ (service request)	トーカからコントローラに対する割り込み要求	
$\overline{\text{EOI}}$ (end or identify)	データの最終バイトであることを知らせる．データの最終バイトを送信する時に"L"レベルにする	

3.2 GP-IB BIOS

GP-IB BIOSは，GP-IBインターフェースボード上のハードウェアを制御するとともに，GP-IBインターフェースを介して接続される各種周辺機器との通信を容易にするための基本プログラムであり，GP-IBインターフェースボードのROM上にすでに用意されています．元来，信号の入出力制御は繁雑なものですが，BIOSでは入出力制御に必要な処理内容をいくつかのBIOSコマンドとして系統化しているので，ユーザも容易に利用することができます．

［GP-IBのINTベクタの初期設定］

GP-IB　BIOSはソフトウエア割り込みで呼び出しますが,そのために割り込みベクタテーブルへの登録をしておく必要があります.割り込みベクタテーブルについては,**表3-1**を参照して下さい.

N_{88}-BASICで使用する場合には，GP-IB用の割り込みベクタコードとして，0 D1Hを使用します．したがって，ユーザは，ベクタテーブルのベクタコード0 D1Hに対応する割り込み先アドレス（セグメントアドレスとオフセットアドレス）をあらかじめ設定しておかなければなりません．割り込み先アドレスに関する情報は，GP-IBインターフェースボードを実装した時点で，CPUアドレスD5400H以降に格納されます．その様子を以下に示します．

相対アドレス	第1バイト	第2バイト	第3バイト	第4バイト	備考
00H	01H	×	×	×	01Hはエントリ数
04H	D1H*2	00H	GP-1B BIOSのオフセットアドレス*1		

*1セグメントアドレス＝D5400H　*2GP-1B BIOSの割り込みベクタコード（N_{88}-BASIC使用の場合）

ユーザは，上記表に基づいてGP-IB　BIOSのオフセットアドレスを読みだし，割り込みベクタテーブル上の所定の位置に，セグメントアドレスとともに記入しなければなりません．

GP-IB　BIOSを利用する場合の手続きを，以下に示します．

① 実行したいGP-IB BIOSコマンドに対応するコマンドコードをレジスタAHにセットする．コマンドの種類によっては，何種類かのレジスタの値をセットしておく必要がある．

② レジスタ設定後，下記命令により，ソフトウェア割り込み（割り込み番号0D1H）を実行すれば，BIOSコマンドが実行される．

```
INT　0D1H
```

(1)初期化コマンド

[機能]

GP-IBインターフェースのハードウェアの初期設定を行う．GP-IB BIOSで使用するワークエリアの初期設定をする．

[割り込みコード]

INT　0D1H

[コマンドコード]

AH←00H

ES←ワークエリアのセグメントアドレス

ワークエリアは下記16バイトで構成される．

ワークエリア円オフセットアドレス	フィールド名称	解説			
		ビット	名称	ビット値＝0	ビット値＝1
0000H	モード	b_7		拡張INT0使用	拡張INT4,5,6使用
		b_6		IFC未受信	IFC受信
		b_5		マスタモード	スレーブモード
		b_4～b_0		マイアドレス	
0001H	アドレスステータス	b_7	CIC	コントローライナクティブ	コントローラアクティブ
		b_6		−	−
		b_5	SPMS	シリアルポール非実行中	シリアルポール実行中
		b_4,b_3		−	−
		b_2	LA	リスナとしてアドレスされていない	されている
		b_1	TA	トーカとしてアドレスされていない	されている
		b_0		−	−
0002H	インターラプトステータス1	b_7,b_6		−	−
		b_5	DET	デバイスリガ受信なし	受信あり
		b_4		−	−
		b_3	DEC	デバイスクリア受信なし	受信あり
		b_2	ERR	送信正常終了	異常終了
		b_1	DO	データ送信要求なし	要求あり
		b_0	DI	データ送信なし	送信あり
0003H	インターラプトステータス2	b_7		−	−
		b_6	SRQI	SRQ受信なし	受信あり
		b_5	LOK	非ロックアウト状態	ロックアウト状態
		b_4	REM	非リモート状態	リモート状態
		b_3		−	−
		b_2	LOCK	LOKビット変化なし	変化あり
		b_1	REMC	REMビット変化なし	変化あり
		b_0	ADSC	CIC,LA,TAビット変化なし	いずれか変化あり
0004H ～ 000FH	作業域				

(2)IFCコマンド

［機能］

$\overline{\text{IFC}}$（インターフェース・クリア）ラインを一定時間アクティブ状態にする．

注）初期化コマンド実行後に，引き続いてIFCコマンドを実行することにより，GP—IBコントローラの状態をアクティブに設定できる．

［割り込みコード］

INT　0D1H

［コマンドコード］

AH←01H

［入力］

BH←アクティブ状態保持時間

注）上記値×100μ secが実際の時間となる．

(3)RENコマンド

［機能］

$\overline{\text{REN}}$（リモート・イネーブル）ラインをアクティブに設定する．

注）初期化コマンド実行後には，RENラインはインアクティブになっている．

［割り込みコード］

INT　0D1H

［コマンドコード］

AH←02H

(4)RENリセットコマンド

［機能］

$\overline{\text{REN}}$（リモート・イネーブル）ラインをインアクティブに設定し，約100μsec待つ．

［割り込みコード］

INT　0D1H

［コマンドコード］

AH←03H

(5)データ送信コマンド

［機能］

GP-IB上に，コマンドおよびデータを送信する．データ送信終了時に送出するデリミタ（CR，LF，CR＋LF，EOI）の形式を設定する．

［割り込みコード］

INT　0D1H

［コマンドコード］

AH←04H

［入力］

ES←コマンドおよびデータ格納域のベースアドレス
SI ←コマンド格納域のオフセットアドレス
BX←コマンド格納域の長さ（単位：バイト）
DI ←データ格納域のオフセットアドレス
CX←データ格納域の長さ（単位：バイト）
AL←デリミタ指定

00H：デリミタ送信不要
01H：デリミタはCR＋LF
02H：デリミタはCR
03H：デリミタはLF
80H：デリミタはEOI
81H：デリミタはCR＋LFかつEOI
82H：デリミタはCRかつEOI
83H：デリミタはLFかつEOI

⑹データ受信コマンド

［機能］

GP-IB上にコマンドを送信した後に，トーカからのデータを受信する．

［割り込みコード］

INT　0D1H

［コマンドコード］

AH←05H

［入力］

ES←コマンドおよびデータ格納域のセグメントアドレス

SI ←コマンド格納域のオフセットアドレス

BX←コマンド格納域の長さ（単位：バイト）

DI ←データ格納域のオフセットアドレス

CX←データ格納域の長さ（単位：バイト）

AL←デリミタ指定

- 81H：受信終了はCR＋LFまたはEOI
- 82H：受信終了はCRまたはEOI
- 83H：受信終了はLFまたはEOI
- 84H：受信終了はEOI

⑺シリアルポール実行コマンド

[機能]

指定したトーカに対して，シリアルポールを実行する.

[割り込みコード]

INT　0D1H

[コマンドコード]

AH←06H

[入力]

ES←トーカ情報リスト領域のセグメントアドレス

DI←トーカ情報リスト領域のオフセットアドレス

CX←トーカ情報リスト領域のエントリの数（トーカの数）

トーカ情報リストの構成を以下に示す.

1エントリにつき，2バイトで構成される.

セグメント内オフセットアドレス	フィールド名称	解説
0000H 0001H	トーカアドレス1 STB1	ステータスバイト } エントリNo.1
0002H 0003H	トーカアドレス2 STB2	ステータスバイト } エントリNo.2
≈	≈	≈ } エントリNo.n ≈

⑻SRQ設定コマンド

[機能]

GP-IBにSRQ（サービスリクエスト）を送信し，GP-IBコントローラからのシリアルポールに応じてSTBを送信する.

[割り込みコード]

INT　0D1H

[コマンドコード]

AH←07H

[入力]

BH←STBコード

シリアルポール時に送信するSTBの値00H～FFH

BL←EOI指定（00H:STB 送信時にEOIを送信する，01H:送信しない）

⑼パラレルポール実行コマンド

［機能］

パラレルポールのライン割り付け,パラレルポールの起動,およびPPRの受信を行う.

［割り込みコード］

INT　0D1H

［コマンドコード］

AH←08H

［入力］

ES←リスナ情報リスト領域のベースアドレス
DI←リスナ情報リスト領域のオフセットアドレス
CX←リスナ情報リスト領域のエントリ数（リスナの数）
BH←パラレルポール起動指定

- 00H：起動しない
- 01H：起動する

BL←PPU指定

- 00H：パラレルポール割り付け前にPPUを送信しない.
- 01H：パラレルポール割り付け前にPPUを送信する.

リスナ情報リストの構成を以下に示す.
1エントリにつき，2バイトで構成される.

セグメント内オフセットアドレス	フィールド名称	解説
0000H 0001H	リスナアドレス1 PPEまたはPPR	}エントリNo.1
0002H 0003H	リスナアドレス2 PPEまたはPPR	}エントリNo.2
≈	≈	}エントリNo.n ≈

⑽PPRモード設定コマンド

[機能]

GP-IBコントローラからのパラレルポールに対する応答（PPR）のモード設定を行う．

[割り込みコード]

INT　0D1H

[コマンドコード]

AH←09H

[入力]

BH←PPRモード

00H：PPRは0

01H：PPRは1

02H：PPRはSRQ送信時に1，SRQ未送信時に0

⑾タイムアウト設定コマンド

[機能]

GP-IBがハングアップしたかどうかを監視する．タイムアウトチェックの時間を設定する．

[割り込みコード]

INT　0D1H

[コマンドコード]

AH←0AH

[入力]

BH←タイムアウト値(単位：秒)．ただし，00Hのときはタイムアウトチェックしない．

注）初期設定コマンド実行後のタイムアウト値は００Ｈである．

⑿チェックSTBコマンド

[機能]

現在保持しているSTBの値とEOI指定状況を通知する.

[割り込みコード]

INT　0D1H

[コマンドコード]

AH←0BH

[出力]

DH←現在保持しているSTBの値

DL←現在保持しているEOIの指定状況（⑻SRQ設定コマンド参照）

4 マウスインターフェース

4.1 マウスインターフェースの概要

PC-98は，オプションでマウスを使用できます．マウス本体底部には球形のローラが付いていて，平面上を移動させたときの移動量を検出する機能があります．マウスの移動に連動させてCRTに表示しているカーソルを移動させるという使い方が典型的な使用形態です．また，マウス本体上部には２つのスイッチがあり，このON/OFF状態は，PC-98から検出できます．

マウスとPC-98との通信の制御を行うためのソフトウェア（マウスBIOSと呼ぶ）は，システムディスク上に用意されています．

4.2 マウスBIOS

マウスBIOS**とは，マウスとPC-98の通信の際に必要な入出力制御を行うためのソフトウェアであり，システムディスク上に用意されています．N_{88}-日本語BASIC(86)のシステムディスクでは下記ファイル名でストアされています．

mouse. cod

マウスBIOSの占有するメモリサイズは約４Kバイトであり，これを機械語領域にロードします．ロードするための手続きのサンプルを以下に示します．

```
CLEAR  &H1F00***
DEF  SEG=&H1F00***
BLOAD  "mouse. cod"
```

次に，マウスBIOSの初期設定を行います．そのための手続きを以下に示します．

---

*　マウスは，マウス本体とマウス用インターフェースボードからなる．F3／M／U2／VF／VM／UV／VX／UXには，マウス用インターフェースが標準実装されているので，マウス本体のみを追加すればよい．

**　BIOS一般については，第３章３および５を参照．

***使用するBASICのシステム・ディスク，メモリ容量によっては値を変更する必要がある．実際の値はBASICユーザーズ・マニュアルを参照．

```
MINIT＝&H100
PARA%＝3
CALL MINIT(PARA%)
```

PARA%の値は，CRTの解像度によって次のように設定します．

PARA%＝ { 3　　高解像カラー（600×400ドット）
　　　　{ 0　　標準カラー　（640×200ドット）

上記手続き実行後には，PARA%にはリターンコードが戻されます．

PARA%＝ { −1　初期設定　正常終了
　　　　{ 0　 異常（マウスインターフェースの不在etc）

マウスBIOSでは，通信の際の入出力制御に必要な処理内容をいくつかのBIOSコマンドとして系統化しているので，ユーザも容易に活用できます．マウスBIOSを利用する場合の手続きを以下に示します．

① 実行したいマウスBIOSコマンドに対応するコマンドコードをレジスタAXにセットする．

　コマンドの種類によっては，何種類かのレジスタの値をセットしておく必要がある．

② レジスタ設定後，下記命令によってソフトウェア割り込み（割り込み番号33H）を実行すれば，BIOSコマンドが実行される．

　INT　33H

　この手続きは，高級言語におけるサブルーチンコールの手続きによく似ています．サブルーチンコールする際に引数を指定しますが，これはBIOSにおけるレジスタ設定に対応しています．

以下では，個々のマウスBIOSコマンドについて説明します．

(1)初期化コマンド

[機能]

カーソル表示，カーソル形状，ミッキー/ドット比，マウス割り込み周期などの初期設定を行う．

[割り込みコード]

INT　33H

[コマンドコード]

AX←0000H

[出力]

AX←マウスの状態を示すパラメータ

0000H：マウス使用不可
FFFFH：マウス使用可

(2)カーソル表示コマンド

[機能]

カーソルをCRTに表示させる．

注）カーソル消去コマンドを実行するまで消えない．

[割り込みコード]

INT　33H

[コマンドコード]

AX←0001H

(3)カーソル消去コマンド

[機能]

カーソルをCRTに表示しなくする．

注）表示しないだけで，カーソルはマウスの動きに応じて移動している．

[割り込みコード]

INT　33H

[コマンドコード]

AX←0002H

(4)カーソル位置検出コマンド

［機能］

カーソルの現在位置を読み取る．マウスの２つのスイッチのON/OFFも検出する．

［割り込みコード］

ＩＮＴ　33H

［コマンドコード］

AX←0003H

［出力］

AX←左側スイッチの状態（0000H：OFF，FFFFH：ON）

BX←右側スイッチの状態（0000H：OFF，FFFFH：ON）

CX←カーソルの水平座標（0～639）

DX←カーソルの垂直座標 ｛カラーモード時　　　（0～199）
　　　　　　　　　　　　　高解像カラーモード時（0～399）

(5)カーソル位置設定コマンド

［機能］

カーソルを指定した位置に移動させる．

［割り込みコード］

INT　33H

［コマンドコード］

AX←0004H

［入力］

CX←カーソルの水平座標（0～639）

DX←カーソルの垂直座標 ｛カラーモード時　　　（0～199）
　　　　　　　　　　　　　高解像カラーモード時（0～399）

⑹左側スイッチON情報読み取りコマンド

［機能］

マウスの左側スイッチが最後に押下されてON状態に切り替わったときのカーソルの座標，および当コマンドを前回実行してから今回実行するまでに左側スイッチがON状態に切り替わった回数を読み取る.

［割り込みコード］

INT　33H

［コマンドコード］

AX←0005H

［出力］

AX←左側スイッチの状態（0000H：OFF，FFFFH：ON）

BX←左側スイッチがON状態に切り替わった回数

CX←最後に左側スイッチがON状態になったときのカーソルの水平座標

DX←最後に左側スイッチがON状態になったときのカーソルの垂直座標

⑺左側スイッチOFF情報読み取りコマンド

［機能］

マウスの左側スイッチが最後に開放されてOFF状態に切り替わったときのカーソルの座標，および当コマンドを前回実行してから今回実行するまでに左側スイッチがOFF状態に切り替わった回数を読み取る.

［割り込みコード］

INT　33H

［コマンドコード］

AX←0006H

［出力］

AX←左側スイッチの状態（0000H：OFF，FFFFH：ON）

BX←左側スイッチがOFF状態に切り替わった回数

CX←最後に左側スイッチがOFF状態になったときのカーソルの水平座標

DX←最後に左側スイッチがOFF状態になったときのカーソルの垂直座標

⑻右側スイッチON情報読み取りコマンド

［機能］

マウスの右側スイッチが最後に押下されてON状態に切り替わったときのカーソルの座標，および当コマンドを前回実行してから今回実行するまでに右側スイッチがON状態に切り替わった回数を読み取る．

［割り込みコード］

INT　33H

［コマンドコード］

AX←0007H

［出力］

AX←右側スイッチの状態（0000H：OFF，FFFFH：ON）

BX←右側スイッチがON状態に切り替わった回数

CX←最後に右側スイッチがON状態になったときのカーソルの水平座標

DX←最後に右側スイッチがON状態になったときのカーソルの垂直座標

⑼右側スイッチOFF情報読み取りコマンド

［機能］

マウスの右側スイッチが最後に開放されてOFF状態に切り替わったときのカーソルの座標，および当コマンドを前回実行してから今回実行するまでに右側スイッチがOFF状態に切り替わった回数を読み取る．

［割り込みコード］

INT　33H

［コマンドコード］

AX←0008H

［出力］

AX←右側スイッチの状態（0000H：OFF，FFFFH：ON）

BX←右側スイッチがOFF状態に切り替わった回数

CX←最後に右側スイッチがOFF状態になったときのカーソルの水平座標

DX←最後に右側スイッチがOFF状態になったときのカーソルの垂直座標

⑽カーソル形状設定コマンド

[機能]

カーソルの形状とカーソルの中心位置を設定する.

[割り込みコード]

INT　33H

[コマンドコード]

AX←0009H

[入力]

BX←カーソル中心点の水平座標（0～15）

CX←カーソル中心点の垂直座標　{ カラーモード時　（0～15）
　　　　　　　　　　　　　　　　　高解像カラーモード時（0～31）

ES←カーソル形状を与えるデータの格納域のセグメントアドレス

DX←カーソル形状を与えるデータの格納域のオフセットアドレス

データの形式は, カラー200モードで16×16ビット, カラー400モードで16×32ビット

⑾カーソル移動量検出コマンド

[機能]

当コマンドを前回実行してから今回実行するまでにマウスが移動した変化を読み取る.

[割り込みコード]

INT　33H

[コマンドコード]

AX←000BH

[出力]

CX←水平方向移動量（-32768～32767）　（単位：ミッキー）

DX←垂直方向移動量（-32768～32767）

⑿ユーザ定義サブルーチンのコール条件設定コマンド

［機能］

ユーザが定義したサブルーチンをマウスの操作によりコールする場合の条件の設定とサブルーチンのアドレスの設定を行う．

［割り込みコード］

INT　33H

［コマンドコード］

AX←000CH

［入力］

CX←コール条件

ビット０：カーソル位置の変化．　　注）ビット０＝LSB
ビット１：左スイッチがONされる．
ビット２：左スイッチがOFFされる．
ビット３：右スイッチがONされる．
ビット４：右スイッチがOFFされる．

上記ビットの値が１のときには，対応する事象が発生したときにコールを実行する．ビットの値が０のときにはコールしない．

ES←ユーザ定義サブルーチンのセグメントアドレス
DX←ユーザ定義サブルーチンのオフセットアドレス

［サブルーチン・エントリ時の状態］

AX←コールの原因となった現象のコード番号

ビット０：カーソルの位置変化
ビット１：左スイッチがONされた．
ビット２：左スイッチがOFFされた．
ビット３：右スイッチがONされた．
ビット４：右スイッチがOFFされた．

BL←左スイッチの状態（0000H：OFF，FFFFH：ON）
BH←右スイッチの状態
CX←カーソル位置の水平座標
DX←カーソル位置の垂直座標

⒀ミッキー／ドット比設定コマンド

［機能］

マウスの移動量とそれに対応するカーソルの移動量の比を設定する．

［割り込みコード］

INT　33H

［コマンドコード］

AX←000FH

［入力］

CX←水平方向のミッキー/ドット比

DX←垂直方向のミッキー/ドット比

⒁水平方向カーソル移動範囲設定コマンド

［機能］

カーソル中心点の水平方向移動範囲を設定する．

［割り込みコード］

INT　33H

［コマンドコード］

AX←0010H

［入力］

CX←水平方向移動範囲の最小値（0～639）

DX←水平方向移動範囲の最大値（0～639）

⒂垂直方向カーソル移動範囲設定コマンド

［機能］

カーソル中心点の垂直方向移動範囲を設定する．

［割り込みコード］

INT　33H

［コマンドコード］

AX←0011H

［入力］

CX←垂直方向移動範囲の最小値 { カラーモード時（0～199） / 高解像カラーモード時（0～399）

DX←垂直方向移動範囲の最大値 { カラーモード時（0～199） / 高解像カラーモード時（0～399）

⒃カーソル表示画面の選択コマンド

［機能］

カーソルを表示させる画面を選択設定できる．

［割り込みコード］

INT　33H

［コマンドコード］

AX←0012H

［入力］

BX←カーソルを表示させる画面の選択コード番号

- 0：プレーン0
- 1：プレーン1
- 2：プレーン2
- 3：プレーン3

5 プリンタインターフェース

5.1 プリンタインターフェースの概要

PC-9801には，セントロニクスインターフェース準拠のプリンタインターフェースが標準装備されています．このプリンタインターフェースのハードウェアには，μPD8255AというLSIを使用しています．このLSIは汎用パラレルインターフェースと呼ばれるもので，制御命令を与えることによって，種々の動作モードに設定して使用することができます．この汎用インターフェースをセントロニクス準拠のインターフェースとして機能させるための基本制御ソフトウェアがPC-98のROM上にすでに用意されていて，プリンタBIOSと呼んでいます．

5.2 プリンタインターフェースのI/O制御命令

プリンタインターフェースの動作状態を制御するために使用しているI/Oポートアドレスは37H，40H，42H，44H，46Hの5種類あります．プリンタインターフェース制御のために準備されている制御命令を**表6-3**にまとめています．表には，各制御命令で使用するI/Oポートアドレスと，そのとき出力される制御データの形式を示しています．

表6-3 プリンタインターフェースのI/Oの制御命令

制御命令	I/Oポート アドレス	I/O	制御データ								機能
			b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0	
ライトモード	46H	OUT	1	0	0	0	0	0	1	0	μPD8255Aの動作モードを設定する
ライトシグナル1	46H	OUT	0	0	0	0	1	1	1	D1	PSTBのON/OFF　D1＝0：OFF 1：ON
ライトシグナル2	46H	OUT	0	0	0	0	0	1	1	D2	IR8のON/OFF　D2＝0：OFF 1：ON
ライトシグナル3	46H	OUT	D1	0	0	0	D2	0	0	0	ライトシグナル1, 2を複合した機能. D1,D2の定義は上記に等しい
ライトデータ	40H	OUT	←――データ――→								プリンタに8ビットデータを送る
リードデータ	40H	IN	←――データ――→								プリンタの動作状態に関するデータを受信する
リードシグナル1	42H	IN	←――各ステータス――→								プリンタの動作状態を読み取る
リードシグナル2	44H	IN	D1	×	×	×	D2	×	×	×	ライトシグナル3の逆動作. μPD8255AのボードCの状態を読み取る
ライトポートC	37H	OUT	0	0	0	0	1	1	0	D0	PSTB用のマスタF/FのON/OFF D3＝0：OFF 1：ON

≡5.3≡ プリンタBIOS

プリンタBIOSは，汎用パラレルインターフェースであるμPD8255AというLSIが，セントロニクス準拠のプリンタインターフェースとして機能するようにするための基本制御プログラムです．データの通信制御は，元来繁雑なものですが，BIOSではユーザが容易に活用できるように，必要となる処理内容をいくつかのBIOSコマンドとして系統化してあります．

プリンタBIOSを利用する場合の手続きを以下に説明します．

① 実行したいプリンタBIOSコマンドに対応するコマンドコードをレジスタAHにセットする．コマンドの種類によっては，何種類かのレジスタの値もセットしておく必要がある．

② レジスタ設定後，下記命令によってソフトウェア割り込み（割り込み番号は1AH）を実行すれば，BIOSコマンドが実行される．

```
INT　1AH
```

この手続きは，高級言語におけるサブルーチンコールの手続きによく似ています．サブルーチンコールする際に引数を指定しますが，これはBIOSにおけるレジスタ設定に対応しています．

以下では，個々のBIOSコマンドについて説明します．

(1)初期化コマンド

[機能]

μPD8255Aおよびステータス情報エリアの初期化を行う.

[割り込みコード]

INT　1AH

[コマンドコード]

AH←10H

[出力]

AH←ステータス情報

MSB							LSB
b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0

b_0＝1：データ送信可能
　　0：データ送信不可能

(2)データ出力コマンド

[機能]

セントロニクス仕様プリンタへ1バイトデータを送信する.

[割り込みコード]

INT　1AH

[コマンドコード]

AH←11H

[入力]

AL←1バイトデータ（JISコード）

[出力]

AH←ステータス情報

MSB							LSB
b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0

b_0＝1：データ送信終了
　　0：データ未送信状態
b_1＝1：タイムアウトになり，データ未送信.
　　0：データ送信完了

⑶ステータス受信コマンド

［機能］

プリンタのステータス情報を要求し，受信する．

［割り込みコード］

INT　1AH

［コマンドコード］

AH←12H

［出力］

AH←ステータス情報

MSB							LSB
b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0

b_0＝1：データ出力可能
　　0：データ出力不可能

⑷複数バイトデータ送信コマンド

［機能］

指定したバッファ上の指定した長さのデータを出力する．

［割り込みコード］

INT　1AH

［コマンドコード］

AH←30H

［入力］

ES←データバッファ先頭部のセグメントアドレス
BX←データバッファ先頭部のオフセットアドレス
CX←出力データ長（バイト）

［出力］

AH←ステータス情報

MSB							LSB
b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0

b_1＝1：データ出力中にタイムアウトが発生，異常終了（未出力データが残存）
　　0：正常終了

BX←タイムアウト発生時の出力データのアドレス（オフセットアドレス）
CX←未出力データ長

第7章

UX & VX

CONTENT

1 概要

本章では，従来機種とは比較的大きな相違のあるPC-9801シリーズVX／UXについての資料を掲載します．

VX／UXシリーズの一番の特徴は，80286CPUを搭載していることでしょう．ただし，80286は従来機種に使用されていた8086CPU，V30とは相違点が多く，従来機と完全な互換性を保つのが難しいため，V30 CPUも搭載してスイッチにより切り替えて使用できるようになっています．

2 80286

80286の特徴は3つあります．

(1) 実行速度が速い

およそV30の約2倍ほどの速さでプログラムを実行します(80286(10MHz)：V30(10MHz)比*)．ただしこれはメモリ操作に限ったことで，ディスクのアクセスなどを伴う場合にはこの限りではありません．

(2) 2つのモードが存在する

80286を高速型8086として利用するモードを「リアル・モード」といいます．リアル・モードでは，8086用に開発されたほとんどのソフトウェアが変更しないで実行できるため，PC-98では主にこのモードで使われます．

80286の本来のモードであり，そのすべての機能を引き出せるモードは「プロテクト・モード」といいます．プロテクト・モードは，マルチタスク・マルチ

---

* CPUのクロックとして選べる周波数は機種により制限がある．

VX0/2/4	V30(8MHz/10MHz)，80286(8MHz)
VX01/21/41	V30(8MHz/10MHz)，80286(8MHz/10MHz)
UX21/41	V30(8MHz)，80286(10MHz)

ユーザを想定したモードですが，このモードを活用するには高度なOS(オペレーティング・システム)が必要になります．

(3)メモリ空間が広い

物理的なメモリエリアを16Mバイトまで持つことができます(ハードディスクなどを用いて，仮想的には1タスクあたり1Gバイトのメモリ空間を持つことが可能)．PC-98ではメインメモリは最大640Kバイトまでしか持つことができませんが，最近のアプリケーションプログラムの高度化によって，640Kバイトでは不足するようになってきました．しかし，プロテクトモードで動作するOSを用いることによって，より多くのメモリを使うことができるようになります．

表7-1 VX/UX I/Oポートアドレス一覧

ポートアドレス																デバイス名	LSI
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
×	×	×	×	0	0	×	×	0	0	0	0	0	×	A_0	0	割込コントローラ・マスタ ＊1	71059
×	×	×	×	0	0	×	×	0	0	0	0	1	×	A_0	0	割込コントローラ・スレーブ ＊1	71059
×	×	×	×	0	0	×	×	0	0	0	A_3	A_2	A_1	A_0	1	DMAコントローラ	8237A
×	×	×	×	0	0	×	×	0	0	1	0	×	×	×	0	カレンダ時計	4990A
×	×	×	×	0	0	×	×	0	0	1	0	×	A_1	A_0	1	DMAバンク	
×	×	×	×	0	0	×	×	0	0	1	1	×	×	A_0	0	RS-232C I/F	8251A
×	×	×	×	0	0	×	×	0	0	1	1	×	×	A_0	1	システムポート	8255A
×	×	×	×	0	0	×	×	0	1	0	0	×	A_1	A_0	0	プリンタI/F	8255A
×	×	×	×	0	0	×	×	0	1	0	0	×	×	A_0	1	キーボードI/F	8251A
×	×	×	×	×	×	×	×	0	1	0	1	×	×	A_0	0	NMI	
×	×	×	×	0	0	×	×	0	1	1	0	A_2	A_1	A_0	0	CRTコントローラ・テキスト	7220A
×	×	×	×	0	0	×	×	0	1	1	0	×	×	×	1	予約	
×	×	×	×	0	0	×	×	0	1	1	1	A_2	A_1	A_0	0	CRTコントローラ	52611
×	×	×	×	0	0	×	×	0	1	1	1	×	A_1	A_0	1	タイマコントローラ	8253
×	×	×	×	×	×	×	×	1	0	0	0	0	0	A_0	0	固定ディスクI/F ＊2	
×	×	×	×	×	×	×	×	1	0	0	0	0	1	×	0	予約	
×	×	×	×	×	×	×	×	1	0	0	0	0	A_1	A_0	1	BRANCH4670	
×	×	×	×	0	0	0	1	1	0	0	0	1	A_1	A_0	0	サウンドボード ＊3	YM2203
×	×	×	×	×	×	×	×	1	0	0	0	1	A_1	A_0	1	ネットワークI/F ＊3	
×	×	×	×	×	×	×	×	1	0	0	1	×	A_1	A_0	0	1MBFDC	765A
×	×	×	×	×	×	×	×	1	0	0	1	0	A_1	A_0	1	CMT I/F ＊3	8251A
×	×	×	×	×	×	×	×	1	0	0	1	1	0	A_0	1	GP-IBスイッチ ＊3	
×	×	×	×	×	×	×	×	1	0	0	1	1	1	×	×	予約	
×	×	×	×	0	0	×	×	1	0	1	0	A_2	A_1	A_0	0	CRTコントローラ・グラフ	7220A
×	×	×	×	0	1	×	×	1	0	1	0	A_2	A_1	A_0	0	EGC制御	
×	×	×	×	0	0	×	×	1	0	1	0	A_2	A_1	A_0	1	文字パターンROM	
×	×	×	×	×	×	×	×	1	0	1	1	0	A_1	A_0	0	通信制御アダプタ	7201
×	×	×	×	×	×	×	×	1	0	1	1	0	A_1	A_0	1	〃	8255A
×	×	×	×	×	×	×	×	1	0	1	1	1	A_1	A_0	1	〃	8253A
×	×	×	×	×	×	×	×	1	0	1	1	A_3	A_2	A_1	A_0	RS-232C拡張I/F	
×	×	×	×	0	0	×	×	1	0	1	1	1	1	1	0	1MB/640KB切換I/O	
×	×	×	×	×	×	×	×	1	1	0	0	0	×	×	0	予約	
×	×	×	×	×	×	×	×	1	1	0	0	1	A_1	A_0	0	640KBFDC	765A
×	×	×	×	×	×	×	×	1	1	0	0	A_2	A_1	A_0	1	GP-IB ＊3	7210
×	×	×	×	×	×	×	×	1	1	0	1	×	×	×	×	ユーザ用	
1	0	1	1	1	1	1	1	1	1	0	1	1	0	0	0	予約	
1	0	1	1	1	1	1	1	1	1	0	1	1	0	1	0	予約	
1	0	1	1	1	1	1	1	1	1	0	1	1	0	1	1	マウス割込周期設定	
0	0	1	1	1	1	1	1	1	1	0	1	1	0	1	1	内部サウンド周波数設定	8253
0	0	1	1	1	1	1	1	1	1	0	1	1	1	1	1	I/Oアドレス77Hと同じ	8253
0	1	1	1	1	1	1	1	1	1	0	1	1	A_1	A_0	1	マウスコントロール	8255A
0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	キーボード(スキャン方式) ＊4	
																｜	
0	0	0	0	0	0	0	0	1	1	1	0	1	0	1	1	〃	
×	×	×	×	×	×	×	×	1	1	1	0	×	×	×	×	ユーザ用	
×	×	×	×	0	0	×	×	1	1	1	1	0	A_1	A_0	0	CPUリセット	
×	×	×	×	0	0	×	×	1	1	1	1	1	A_2	A_1	A_0	NDP	80287

＊1 VX0/2/4はμPD71059
＊2 VX4/VX41は内蔵，他はオプション
＊3 オプション
＊4 BASICのINP関数でのみ有効(機械語レベルではユーザ使用可)

3 拡張されたI／Oポート

VX/UXシリーズでは，80286の搭載などにともなって，システムの状態を読み出すI/Oポートの一部が拡張されました．ここでは，その変更部分のみを取り上げて解説します．

(1)マウスI／F付属部分

マウスI／Fの8255AのポートB，Cから本体全面のディップSWのうちの4つの設定を読み出すことができます．すべてのビットは，0＝ON，1＝OFFです．

ポートB(7FDB) bit 6 ：SW3-6の状態（RAM最上位128K切り離し）
ポートC(7FDD) bit 0 ：SW1-5の状態（RS-232C同期モード）
bit 1 ：SW1-6の状態（RS-232C同期モード）
bit 2 ：SW3-8の状態（CPUの切り替え）

BASICからこの値を読む場合は，INP関数を使います．

例：IF (INP(&H7FDD)AND 4)＝0 THEN PRINT "80286" ELSE PRINT "V30"

なお，古いマウスドライバを使った後では正しく読めない場合があります．このときには，OUT &H7FDF，&H93をあらかじめ実行しておくと正しく読むことができます．

(2)プリンタI／F付属部分

プリンタI／Fの8255AのポートBからは，CPUの動作を読み出すことができます．

ポートB(0042) bit 1 ：動作中のCPU種別（0＝80286，1＝V30）
bit 5 ：動作中のCPUクロック（0＝10MHz，1＝8MHz）

VX0/2/4の80286のCPUクロックは8MHzに固定されています．このため，

VX0/2/4で80286を使っているときには，本体全面のクロック切り替えスイッチが10MHzでも，ポートBのbit5は必ず1になります．

BASICからこの値を読む場合は，INP関数を使います．

例：IF(INP(&H42) AND &H20)＝0 THEN PRINT "10MHz" ELSE PRINT "8MHz"

表7-2に示した数のJMP　$+2*3をI／Oアクセス命令間に入れれば，PC-9801VX／UXの80286動作時のどのクロック状態でも十分なウェイトが得られます．

表7-2 I/O命令連続アクセス時のウエイト挿入数一覧

周辺LSI名	IN→IN	OUT→OUT	OUT→IN	IN→OUT
8255A-5　PPI	1	1	1	1
8253-5　タイマ	1	1	1	1
8251A USART　モード初期化	0	3	0	0
ライトデータ同期	0	4	0	0
ライトデータ非同期	0	7	0	0
8259A/71051　PIC	0	1	1	0
765AC　FDC	0	0	0	0
7220A　GDC(グラフ)＊1 ＊2	1	2	2	1
7220A　GDC(テキスト)＊1	0	1	1	0
8237-5　DMAC	1	1	1	1
7210　GP-IB	0	1	1	0

＊1　高解像度CRT，スーパーインポーズしない場合

＊2　GDCクロック2.5MHzの場合
GDCクロックが5MHzのときは，GDC（テキスト）と同じ

＊3　JMP　$+2のオペコードは，EB　00です．
つまり，JMP　$+2の次の番地にジャンプする，という意味です．モニタやデバッガから入力する場合には，この表記が使えません．この場合，例えば100番地にジャンプ命令を書くときには，JMP　102　とすれば同じです．
参考までに，JMP　$+2の命令実行時間はNOPを4個実行するのより，わずかに速い程度です．つまり，NOP 4個で置き換えることも可能です．

索引

あ

か

さ

た

な

は

ま

や

ら

わ

A～Z

●著者略歴
東京工業大学　電子計算機愛好会
稲福　鑒
江口正浩
堀田権一

小高　輝真(こだかてるまさ)

98ハードに強くなる本 II

昭和63年 2 月25日　初版 第 1 刷発行
平成 3 年10月15日　初版 第12刷発行

著　者　東京工業大学　電子計算機愛好会
　　　　小高　輝真
発行者　片岡　巌
発行所　株式会社技術評論社
　　　　東京都新宿区愛住町 8 番地 8
　　　　電話　03(3225)2300(代) 営業部
　　　　　　　03(3225)3293(代) 編集部
印刷／製本　加藤文明社

定価はカバーに表示してあります

ISBN4-87408-931-3 C3055
Printed in Japan